(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENT- UND MARKENAMT

(12) Übersetzung der europäischen Patentschrift

(87) EP 0 578 262 B1

(10) DE 693 21 045 T2

(51) Int. Cl.[6]: G 06 F 3/12

(21) Deutsches Aktenzeichen: 693 21 045.1
(86) Europäisches Aktenzeichen: 93 111 048.0
(86) Europäischer Anmeldetag: 9. 7. 93
(87) Erstveröffentlichung durch das EPA: 12. 1. 94
(87) Veröffentlichungstag der Patenterteilung beim EPA: 16. 9. 98
(47) Veröffentlichungstag im Patentblatt: 18. 2. 99

(30) Unionspriorität:

911767	10. 07. 92	US
912098	10. 07. 92	US
976355	16. 11. 92	US

(73) Patentinhaber:
Microsoft Corp., Redmond, Wash., US

(74) Vertreter:
Grünecker, Kinkeldey, Stockmair & Schwanhäusser, Anwaltssozietät, 80538 München

(84) Benannte Vertragstaaten:
AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LI, LU, MC, NL, PT, SE

(72) Erfinder:
Dennis, Stephen V., Bothell, Washington 98011-5453, US; Fluegel, Steven J., Bellevue, Washington 98007, US; Gerlach, Brett, C., Bellevue, Washington 98007, US; Flagg, Robert C., Redmond, Washington 98052, US

(54) Verfahren und System für dynamische Drucker "Time out"

DE 693 21 045 T2

DE 693 21 045 T2

EPA 93 111 048.0

Technisches Gebiet

Die vorliegende Erfindung betrifft ein System und ein Verfahren zum Drucken mit Computersystemen.

Hintergrund der Erfindung

Computersysteme sind sehr nützlich beim Zusammenstellen und Verarbeiten von großen Datenmengen. Moderne Computersysteme enthalten umfassen Grafikfähigkeiten, die das Anzeigen und Drucken von Grafikbildern erlauben. Das Drucken eines Textes und/oder von Grafikbildern erfordert die Umwandlung von Daten von dem auf einem Hostcomputer verwendeten Format zu dem Format, das von dem besonderen ausgewählten Drucker verwendet wird. Typischerweise werden die Computerdaten in eine Bitmapdatei umgewandelt, in der jedes Bit einen Punkt auf der gedruckten Seite wiedergibt. Die Bitmap wird gewöhnlich im Hostcomputer erzeugt und in einem komprimierten Datenformat zum Drucker übertragen. Die komprimierte Bitmap wird im Drucken dekomprimiert und zum Druckermechanismus übertragen, d.h. zu dem Teil des Druckers, der die Papierbewegung und den Toner steuert, zu dem mechanischen Antriebssystem, das das Papier bewegt, und zu der elektronischen Schaltung, die den Druckvorgang steuert. Der Druckermechanismus empfängt die Bitmapdaten und wandelt sie in die entsprechenden Spannungen zum Erzeugen eines gedruckten Bildes um.

Das gedruckte Bild setzt sich aus einzelnen Punkten zusammen, die als Pixel bezeichnet werden. Bei einem typischen Laserdrucker können 300, 600 oder mehr Pixel pro Zoll vorgesehen sein. Jedes Pixel wird gewöhnlich durch ein einziges Datenbit im Druckerspeicher wiedergegeben. Wenn der Laserdruckermechanismus eine Zeile scannt, werden die Datenbits für diese Zeile gelesen und der Laserstrahl wird je nach dem Logikpegel des an dieser Speicherstelle gespeicherten Datenbits

an- oder abgeschaltet. Bei einer Auflösung von 300 Pixel pro Zoll braucht ein Drucker ungefähr ein Megabyte Speicher, um die Bitmapdatei für eine ganze Seite zu speichern. Einige Laserdrucker enthalten große Speicher und operieren in einem Seitenmodus, was bedeutet, daß der Drucker eine ganze Seite von Daten in der Form einer Bitmap speichern kann. Wegen der großen Speichermenge, die zum Speichern einer ganzen Seite erforderlich ist, verwenden einige Drucker einen Bandmodus, um die erforderliche Speichermenge zu reduzieren. Ein Drucker mit Bandmodusfähigkeiten teilt die Seite in eine Anzahl von horizontalen Segmenten, die als Bänder bezeichnet werden. Der Drucker nimmt jeweils nur die Bitmap-Daten für ein Band an, wodurch die Speicheranforderungen reduziert werden. Wenn der Drucker die Daten für das erste Band verarbeitet hat, kann er die Daten für das zweite Band annehmen usw. Wenn die Seite zum Beispiel in vier Bänder unterteilt wird, werden die Speicheranforderungen um einen Faktor von vier reduziert, da der Drucker lediglich die Bitmapdaten für ein Viertel der Seite zu speichern braucht.

Bei der Entwicklung eines Computer-Drucker-Systems müssen zwei Grundzielsetzungen erreicht werden. Die erste Zielsetzung betrifft die Geräteunabhängigkeit. Um den Drucker von den Systembeschränkungen eines bestimmten Hostcomputers unabhängig zu machen, entwickeln die Hersteller und Programmierer Druckertreiber, die möglichst universal sind. Wenn eine echte Geräteunabhängigkeit gegeben ist, braucht der Hostcomputer nicht zu wissen, welcher Typ von Drucker mit der Druckerschnittstelle verbunden ist. Der Hostcomputer überträgt die Daten typischerweise über eine Hardwareschnittstelle wie einen seriellen I/O-Port oder einen parallelen Drucker-I/O-Port zum Druckersystem. Die Computer-Drucker-Systeme des Standes der Technik versuchen eine Geräteunabhängigkeit zu erreichen, indem sie das Wissen des Hosts über den besonderen mit ihm verbundenen Drucker minimieren und sich statt dessen auf einen eher abstrakten Datenstrom verlassen. Das hat jedoch zur Folge, daß die Verarbeitung des Datenstroms häufig auf eine ineffiziente Geschwindigkeit verlangsamt wird, so daß der Durchsatz leidet.

Zum Beispiel kann ein Hostcomputer einen ersten Fontsatz laden, der für mehrere Seiten eines Dokuments gebraucht wird. Typischerweise laden die Computersysteme des Standes der Technik einen vollständigen Font, auch wenn nur einige wenige Zeichen für eine bestimmte Druckaufgabe benötigt werden. Der Hostcomputer lädt einen vollständigen zweiten Fontontsatz, wenn der zweite Fontsatz (oder ein Teil des zweiten Fontsatzes) gebraucht wird. Der zweite Fontsatz kann in den Speicherraum geschrieben werden, der durch den ersten Fontsatz besetzt ist, auch wenn Platz im Druckerspeicher vorhanden ist, um den ersten Fontsatz für die Verwendung während des Druckens von darauffolgenden Seiten zu behalten. Es ist keine Kommunikation vom Drucker zum Hostcomputer vorgesehen, um den Hostcomputer über den aktuellen Status der Druckerressourcen zu informieren.

Die zweite Zielsetzung betrifft die optimale Leistung des Druckvorgangs. Laserdrucker reichen von einfachen autonomen Druckern, die mit einem einzigen Computer verbunden sind und jeweils eine oder zwei Papierseiten verarbeiten, bis zu hochentwickelten Druckern mit mehreren Papierschächten und komplexen Papierwegen, die mit einem Computernetz verbunden sind und viele Seiten für mehrere Benutzer verarbeiten. Computersysteme müssen in der Lage sein, effektiv mit jedem Typ von Drucker arbeiten zu können.

Unglücklicherweise, können die erste und die zweite Zielsetzung miteinander in Konflikt geraten. Der Kompromiß bei dem Versuch, eine universelle Kompatibilität zu erreichen, resultiert in häufig sehr langsamen Geschwindigkeiten bei der Verarbeitung der Daten. Weiterhin verfügen die Hostcomputer über ein gewisses Wissen über den Drucker, mit dem sie verbunden sind. Bei dem Versuch, beide Zielsetzungen zu erreichen, erreichen gegenwärtige Computer-Drucker-Systeme ironischerweise keine der beiden Zielsetzungen. Der Hostcomputer weiß, mit welchem Typ von Drucker er verbunden ist, und trotzdem resultiert der Ansatz eines „universellen" Druckertreibers in einem langsamen, ineffizienten System, bei dem der Hostcomputer und der Drucker häufig wertvolle Berechnungszeit dafür verbrauchen, Konflikte zu lösen, die nicht auftreten dürfen. So müssen sie zum Beispiel eine

Seitenfehlerkorrektur durchführen, anstatt nützliche Aufgaben auszuführen. Die Computer-Drucker-Systeme des Standes der Technik halten zum Beispiel die Bitmapdatei für eine ganze Seite zurück, bis die gedruckte Seite den letzten Papierstau-Sensor im Laserdrucker passiert. Wenn ein Papierstau auftritt, sind die Daten weiterhin verfügbar und die Seite kann schnell erneut gedruckt werden. Papierstaus während des Druckvorgangs treten jedoch relativ selten auf. Wenn der Druckermechanismus mit dem Drucken einer Seite beginnt, dauert es ungefähr zehn Sekunden, bis die Seite den letzten Papierstau-Sensor passiert. Der Gesamt-Druckvorgang wird beträchtlich verlangsamt, wenn bei jeder Seite zusätzlich zehn Sekunden darauf gewartet werden muß, daß der letzte Papierstau-Sensor passiert wird, bevor die Bitmapdatei aus dem Drucker gelöscht und die nächste Seite verarbeitet werden kann.

Die Systeme des Standes der Technik verzögern auch das Einziehen des Papiers in den Druckermechanismus, bis die ganze Seite beschrieben ist, da die Wahl der Papiergröße zu einem beliebigen Zeitpunkt während der Seitenbeschreibung gemacht werden kann. Der Hostcomputer kann zum Beispiel die Beschreibung einer ganzen Seite übertragen, wobei die letzte Beschreibungszeile die Wahl der Papiergröße enthalten kann. Es besteht keine Notwendigkeit das Einziehen des Papiers zu verzögern, wenn der Benutzer die Papiergröße zu Beginn der Seitenbeschreibung auswählt. Der Benutzer kennt allgemein die Papiergröße und den Druckmodus (d.h. das einseitige oder doppelseitige Drucken), bevor der Druckvorgang beginnt. Deshalb verschwenden die Systeme des Standes der Technik wertvolle Zeit, indem sie eine unnötige Option bereitstellen.

Die gegenwärtige verwendeten Druckersprachen haben sich aus Druckersprachen entwickelt, die mit Punktmatrixdruckern verwendet wurden. Es werden zwar immer noch Punktmatrixdrucker verwendet, die Verwendung von Laserdruckern ist jedoch verbreitet und nimmt zu. Mit dem zunehmenden Verwendung von Laserdruckern wurde versucht, kleinere Modifikationen bei den an den langsameren Punktmatrixdruckern orientierten Druckersprachen vorzunehmen.

Diese evolutionäre Ansatz nützt jedoch nicht den Vorteil der potentiellen Berechnungsleistung die in Laserdruckern vorhanden ist.

Die Systemarchitektur der Computer-Drucker-Systeme des Standes der Technik hat sich wenig verändert, obwohl sich die Druckerhardware von „dummen" Druckern zu hochentwickelten, von Mikroprozessoren gesteuerten Laserdruckern entwickelt hat. Das in Fig. 1 gezeigte typische Computer-Laserdrucker-System des Standes der Technik weist eine Einrichtung im Drucker auf, die als Parser bekannt ist. Der Parser nimmt Datenbytes vom Hostcomputer an und organisiert die Datenbytes zu Tokens. Tokens sind Datenströme, die in einem bedeutungsvollen lexikalischen Kontext assoziiert sind. Zum Beispiel kann ein Datenstrom eine binäre Bitmap sein, die in einem komprimierten Format übertragen wird. Die binären Daten sind gewöhnlich durch einen Header und einen Trailer begleitet, der dem Parser mitteilt, wie die Daten zu verarbeiten sind. Header und Trailer werden als ASCII-Bytes übertragen, die jeweils durch den Parser verarbeitet werden müssen. Der Parser muß jeweils ein Byte der ASCII-Datenbytes annehmen und verarbeiten. Das hat zur Folge, daß der Parser einen Flaschenhals für den effizienten Datenfluß in einem Computer-Drucker-System darstellt.

Der Parser verarbeitet jedes durch den Drucker empfangene Datenbyte und erzeugt eine Anzeigeliste in einem Speicher im Drucker. Die Anzeigeliste ist nach der Lokation des Objektes auf der Seite geordnet. Bitmaps in der Anzeigeliste sind allgemein in nicht komprimiertem Format gespeichert. Andere Objekte wie Text sind eher kurz. Deshalb erfordert ein einziges, einfaches Rechteck, das entlang der Ränder der Seite verläuft, ein Megabyte Speicherplatz. Ein Imager übersetzt die Anzeigeliste in eine für den Druckermechanismus geeignete Bitmapdatei. Die Bitmapdatei wird in einem Bildpuffer gespeichert und zum Druckermechanismus übertragen.

Computer-Drucker-Systeme des Standes der Technik sind weiterhin deshalb ineffizient, weil die Seiten häufig in einer ineffizieten Reihenfolge verarbeitet werden. Wenn der Drucker im doppelseitigen Modus arbeitet (die Vorder- und die Rückseite

eines Blattes werden bedruckt), erfordert der innerhalb des Druckers genommene Papierweg, daß die Rückseite vor der Vorderseite gedruckt wird. Die Computer-Drucker-Systeme des Standes der Technik erfordern jedoch, daß sie Vorderseite vor der Rückseite verarbeitet wird. Das bedeutet, daß die Vorderseite vollständig verarbeitet und im Druckspeicher als eine Bitmapdatei gespeichert werden muß. Dann wird die Rückseite vollständig verarbeitet und zum Druckermechanismus gesendet. Die Philosophie der Systeme des Standes der Technik nimmt an, daß der Benutzer erwartet, daß die Vorderseite zuerst verarbeitet wird. Der Benutzer erwartet jedoch nur, daß die Seiten in der richtigen Reihenfolge im Druckerschacht erscheinen, wenn das Dokument vollständig gedruckt ist. Es besteht kein praktischer Grund dafür, daß der Hostcomputer die Seiten in einer anderen Reihenfolge verarbeiten sollte als in der Reihenfolge, in der die Seiten tatsächlich durch den Drucker verarbeitet werden.

Wie oben bemerkt, benützen die Systeme des Standes der Technik nicht die potentielle Berechnungsleistung, die bei modernen Laserdruckern zur Verfügung steht. Die dummen Drucker des alten Aufbaus stellten nicht mehr als einen Datenpuffer und einen Druckermechanismus dar. Die Datenverarbeitung wurde ausschließlich durch den Hostcomputer vorgenommen, wobei der Drucker die Punktmatrixdaten druckte. Moderne Laserdrucker werden durch Mikroproessoren gesteuert und verfügen über eine Berechnungsleistung, die sogar der des Hostcomputers gleichkommen kann. Die Systeme des Standes der Technik tendieren dazu, den Drucker weiterhin als einen dummen Drucker zu behandeln, der über keine Fähigkeiten bezüglich der Verarbeitung von Daten verfügt. Der Grund dafür ist zum Teil, eine Geräteunabhängigkeit zu erreichen, wie oben beschrieben wurde. Andere Systeme des Standes der Technik übergeben dem Drucker die Verantwortung für praktisch die gesamte Datenverarbeitung. Das Ergebnis ist, daß die Berechnungsleistung des Hostcomputers und des Druckers nicht effektiv genützt wird, so daß der gesamte Druckvorgang auf eine ineffiziente Geschwindigkeit verlangsamt wird.

Da nur wenig Kommunikation zwischen dem Hostcomputer und dem Drucker stattfindet, können sich der Hostcomputer und der Drucker die Aufgabe der

Datenverarbeitung nicht effizient teilen. Die Hostcomputer des Standes der Technik führen praktisch die gesamte Datenverarbeitung oder praktisch nichts der Datenverarbeitung aus. Zum Beispiel setzt der Hostcomputer des Standes der Technik eine Zeitauslösung unabhängig von der Datenverarbeitung im Drucker oder in einem anderen Peripheriegerät, mit dem der Drucker verbunden ist. Die Zeitauslösung entspricht der Zeitspanne, für die der Hostcomputer erwartet, nicht mit dem Drucker oder dem anderen Peripheriegerät kommunizieren zu können. Im Falle eines Druckers ist der Hostcomputer außer Kommunikation, während der Drucker Bitmaps aufbereitet oder in anderer Weise Daten verarbeitet. Der Hostcomputer verfügt jedoch über keine Möglichkeit, zu wissen, wie lange der Drucker tatsächlich für die Verarbeitung bestimmter Daten braucht. Der Hostcomputer kann also nicht wissen, wie lange die Zeitauslösung ist. Der Hostcomputer kann lediglich einen fixen Wert für alle Aufgaben auswählen und annehmen, daß ein Fehler aufgetreten ist, wenn das Peripheriegerät länger für die Verarbeitung der Daten braucht.

Die Versuche, eine Geräteunabhängigkeit und einen universellen Betrieb mit allen Typen von Druckern zu erreichen, hat ein ineffizientes Drucken zur Folge, bei der potentielle Berechnungsleistung nicht ausgeschöpft wird und bei dem Ressourcen verschwendet werden, da der Hostcomputer und der Drucker nicht effektiv miteinander kommunizieren. Deshalb sollte beachtet werden, daß ein deutlicher Bedarf für ein Computer-Drucker-System besteht, daß eine effektive Kommunikation zwischen dem Hostcomputer und dem Drucker erlaubt und die Nutzung von Ressourcen maximiert.

EP-A-0474153 gibt eine Ausgabevorrichtung an, bei der auf der Basis der Eingabedaten eine angenommene Ausführungszeit in Übereinstimmung mit der Auflösung berechnet wird und verwendet wird, um eine Ausgabeauflösung so hoch wie möglich für die Erzeugung innerhalb der Druckausführungszeit zu setzen.

Zusammenfassung der Erfindung

Das System der vorliegenden Erfindung ist in einem Hostcomputersystem implementiert, das einen Hostressourcenspeicher zum Speichern von Daten und einen Ressourcenassembler umfaßt, der gespeicherte Daten untersucht und die Echtzeitkosten für die Verarbeitung der Daten in einem Peripheriegerät bestimmt. Der Ressourcenassembler kann die tatsächliche Verarbeitungszeit genau bestimmen und setzt eine Zeitauslösung, die wenigstens so lang ist wie die bestimmte Verarbeitungszeit. Die Zeitauslösung kann von einer Task zu einer anderen geändert werden.

In einer Ausführungsform ist der Hostcomputer mit einem Drucker verbünden und kann die Verarbeitungszeit für jede Seite genau bestimmen. Die tatsächliche Verarbeitungszeit überschreitet die Zeitspanne von dreißig Sekunden nicht, kann aber weniger als dreißig Sekunden betragen. Wenn die Verarbeitungszeit mit weniger als dreißig Sekunden bestimmt wird, kann der Hostcomputer eine Zeitauslösung bestimmen, die wenigstens so groß ist wie die bestimmte Verarbeitungszeit. Die Zeitauslösung kann von Seite zu Seite je nach der durch den Ressourcenassembler bestimmten Verarbeitungszeit geändert werden.

Wenn der Hostcomputer und die Peripherie mit einem Netz verbunden sind, muß der Hostcomputer keine Kontrolle über die Zeit üben, zu der die Daten zur Peripherie geleitet werden. Der Netzserver nimmt Daten vom Hostcomputer an und gibt sie an die Peripherie weiter. Deshalb muß der Netzserver die Zeitauslösung verfolgen. In einer Ausführungsform wird dem Netzserver die maximale Verarbeitungszeit gegeben, die durch den Ressourcenassembler bestimmt wird, um sie als Zeitauslösung zu verwenden. In der Ausführungszeit, in der der Hostserver über ein Netz mit dem Drucker verbunden ist, wird dem Netzserver ein Maximum von dreißig Sekunden als Zeitauslösung gegeben.

Kurzbeschreibung der Zeichnungen

Fig. 1 stellt einen typisches Computer-Laserdrucker-System des Standes der Technik dar,

Fig. 2 ist ein Funktionsblockdiagramm des Computer-Drucker-Systems der vorliegenden Erfindung,

Fig. 3A stellt eine Möglichkeit dar, in der ein unbegrenztes Dokument wiedergegeben werden kann,

Fig. 3B stelle eine Möglichkeit dar, in der das unbegrenzte Dokument von Fig. 3A in auf begrenzte wiedergegeben werden kann,

Fig. 3C stellt eine alternative Konfiguration des begrenzten Dokuments von Fig. 3B dar,

Fig. 4 stellt einige Lasten-Balance-Optionen für die vorliegende Erfindung dar,

Fig. 5 ist ein Funktionsblockdiagramm des Kostenmetrik-Systems der vorliegenden Erfindung,

Fig. 6 ist ein Funktionsblockdiagramm des Timers, der in der vorliegenden Erfindung verwendet wird.

Ausführliche Beschreibung der Erfindung

Die vorliegende Erfindung stellt einen revolutionären Ansatz für die Computer-Drucker-Interaktion dar. Sie erlaubt die kooperative Anstrengung des Hostcomputers und des Druckers, um die Druckgeschwindigkeit des Druckvorgangs erheblich zu erhöhen und um die Rückkehr-zur-Anwendung-Zeit zu vermindern. Die Rückkehr-zur-Anwendung-Zeit entspricht der Zeit, die der Hostcomputer braucht, um einen Druckjob zu verarbeiten und um zu dem Anwendungsprogramm zurückzukehren, das das Drucken initiiert hat. Viele Computer-Drucker-Systeme des Standes der Technik sind für das Ausführen eines Computercodes und weniger für das Drucken von Seiten geschaffen. Das heißt, daß der Drucker einen Code empfängt, diesen übersetzt und ausführt, wobei die Seite als ein Nebenprodukt der Codeausführung gedruckt wird. Viele Systeme des Standes der Technik führen eine große Codemenge aus, um eine einzelne Seite zu drucken.

Wie oben beschrieben, stellen die Systeme des Standes der Technik keinen effizienten Dialog zwischen dem Hostcomputer und dem Drucker her. Deshalb werden die hochentwickelten Fähigkeiten des modernen Druckers nicht genutzt. Im Gegensatz dazu, nutzt die vorliegende Erfindung die Berechnungsleistung im Drucker und ist dafür ausgebildet, eine freie Kommunikation zwischen dem Hostcomputer und dem Drucker zu erlauben. Die vorliegende Erfindung betrachtet den Hostcomputer und den Drucker als zwei Teile desselben „Systems", wobei beide Teile miteinander zusammenarbeiten können, um Dokumente effektiv zu drucken. In dem oben angegebenen Beispiel, wo zwei Fontsätze erforderlich sind, um ein Dokument zu drucken, weiß der Computer der vorliegenden Erfindung, daß der Drucker den ersten Fontsatz im Druckerspeicher zurückhalten kann. Weiterhin betrachtet der Hostcomputer des Standes der Technik typischerweise nur die aktuell verarbeitete Seite und blickt nicht voraus, um zu bestimmen, ob der erste Fontsatz vielleicht in der Zukunft gebraucht werden könnte und im Druckerspeicher behalten werden sollte. Als Folge muß der Hostcomputer des Standes der Technik wiederholt den ersten Fontsatz laden, wenn er für das Drucken von mehreren Seiten gebraucht wird. (Dabei kann er den zweiten Fontsatz im Druckvorgang löschen.) Es ist zu beachten, daß einige Systeme aus dem Stand der Technik über grobe voarausblickende Fähigkeiten verfügen, wobei diese jedoch sehr begrenzt sind und den Speicher nicht effizient nutzen. Der Host des Standes der Technik weiß nicht immer sicher, wieviel Speicher verfügbar ist und muß deshalb sehr konservativ in der Verwendung des Druckerspeichers sein. Im Gegensatz dazu, blickt das vorliegende System auf die Druckaufgabe voraus, um zu bestimmen, ob der erste Fontsatz oder eine andere Ressource im Druckerspeicher behalten werden sollte, und um zu bestimmen, wann der erste Fontsatz nicht mehr gebraucht wird und aufgegeben oder aus dem Speicher gelöscht werden kann. Weiterhin erstellt das System der vorliegenden Erfindung Untersätze von Zeichenfonts, wenn nur ein Teil eines Fonts erforderlich ist, was die Verwendung der verfügbaren Ressourcen maximiert. Auf diese Weise verwendet die vorliegende Erfindung einen Ressourcen-orientierten Ansatz für das Drucken.

Die vorliegende Erfindung nimmt auf Details des Betriebs eines Laserdruckers Bezug, wobei jedoch zu beachten ist, daß das System und das Verfahren der vorliegenden Erfindung für jede Markierungstechnologie anwendbar ist, d.h. also für Laser-, Anschlag-, Sublimations-, Tintenstrahldrucksysteme oder ähnliche.

Eine Ressource ist etwas im Computer-Drucker-System, das Speicher besetzt oder für das Drucken eines Dokuments erforderlich ist. Das Dokument kann vollständig unter Verwendung der Ressourcen beschrieben werden. Der Begriff Ressourcen wird weiter unten ausführlicher erläutert. In Übereinstimmung mit den Prinzipien der vorliegenden Erfindung ist es die Zielsetzung, das Dokument schnell zu drucken und dem Hostcomputer zu erlauben, schnell innerhalb von einer minimalen Zeitspanne zum Anwendungsprogramm zurückzukehren. Dies wird bewerkstelligt, indem eine offene Kommunikation zwischen dem Hostcomputer und dem Drucker erlaubt wird, so daß die Anforderungen jedes Teils des Systems und der für die Bewerkstelligung der Aufgabe verfügbaren Ressourcen für die anderen Teile des Systems bekannt sind. Der gesamte Druckvorgang ist schneller als bei Systemen des Standes der Technik, zum Teil deswegen, weil die Berechnungsleistung und der verfügbare Speicher des Hostcomputers und des Druckers mit ins Spiel gebracht werden.

Die vorliegende Erfindung kann mit einem Computer-Drucker-System verwendet werden, das lediglich über unidirektionale Kommunikationsfähigkeiten verfügt und eine Statusleitung vom Drucker zum Hostcomputer umfaßt, um anzugeben, daß der Drucker beschäftigt ist. Andere Computer oder Drucker können bidirektional kommunizieren, wobei sie aber keine volle bidirektionale Kommunikation mit einer Datenrate unterstützen, wie sie für die vorliegende Erfindung erforderlich ist, oder eine vollwertige bidirektionale Kommunikation behindern, weil sie nicht schnell genug zwischen den Richtungen hin und her schalten können. Die vorliegende Erfindung versucht, eine bidirektionale Kommunikation herzustellen, wobei sie jedoch auf eine unidirektionale Kommunikation zurückgreift, wenn entweder der Computer oder der Drucker keine bidirektionale Kommunikation unterstützt, weil die Latenzzeit so lang ist, daß er eine vollwertige bidirektionale

Kommunikation unterstützen könnte. Viele Computer-Drucker-Systeme weisen jedoch eine volle bidirektionale Kommunikation zwischen dem Hostcomputer und dem Drucker auf. Wenn ein Computersystem über bidirektionale Fähigkeiten verfügt, bietet die vorliegende Erfindung verbesserte Fehlerkorrekturfähigkeiten und die Fähigkeit, je nach dem besonderen Druckauftrag einige der Funktionen zwischen dem Drucker und dem Hostcomputer hin und her zu verlagern. Diese „Lasten-Balance" erlaubt sogar noch größere Druckgeschwindigkeiten, indem sie erlaubt, daß der Druckauftrag durch den Teil des Computer-Drucker-Systems gehandhabt wird, der den Auftrag am effektivsten handhaben kann.

Wie zuvor erläutert, kann eine Ressource praktisch alles sein, was zum Drucken eines Dokuments erforderlich ist. Dazu können Fontsätze, Glyphensätze, Punktetabellen, Brushes und andere benutzerdefinierte Grafikbilder und auch Daten gehören, die die Seite selbst beschreiben. Ein „Fontsatz" ist ein Satz von ASCII-Zeichen, der gewöhnlich als Bitmap gespeichert wird und einen bestimmten Schrifttyp wie Times-Roman, Helvetica, Courier oder einen ähnlichen definiert. Einige Drucker weisen Fontsätze auf, die in Nur-Lese-Speicher-(ROM)-ICs im Drucker gespeichert sind, während andere Computer-Drucker-Systeme „weiche Fonts" verwenden, die als Bitmapdateien im Hostcomputer gespeichert sind und bei Bedarf in den Direktzugriffspeicher (RAM) des Druckers geladen werden. Einige Fonts erlauben eine größere Flexibilität beim Drucken, da die Fonts allgemein auf einer Platte im Hostcomputer gespeichert sind und deshalb nicht permanent Speicherplatz im Drucker besetzen. Wieder andere Computer-Drucker-Systeme verwenden ein Font-Skalier-Verfahren, wie etwa TrueType-Fonts, wobei die Fonts nicht als Bitmapdatei gespeichert sind. Statt dessen werden die Fonts als ein Satz von Gleichungen beschrieben, die Linien und Kurven von Zeichen für jeden Fonttyp definieren. Der Hostcomputer oder der Drucker verwendet diese Gleichungen, um ein bestimmtes Fontzeichen mit einer bestimmten Punktgröße zu erzeugen. Der Vorteil des Font-Skalier-Verfahrens ist darin gegeben, daß ein einziger Satz von Gleichungen verwendet werden kann, um einen Fonttyp für alle Punktgrößen zu beschreiben, währen die als Bitmaps gespeicherten Fonts nur für eine einzige Punktgröße verwendet werden können. Zum Beispiel werden Times-Roman 4,

Times-Roman 6, Times-Roman 8 und Times-Roman 10 als vier separate Fonts betrachtet, wobei jede eine separate Bitmapdatei erfordert, um den bestimmten Font zu beschreiben. Im Gegensatz dazu verwendet das Font-Skalier-Verfahren einen einzigen Satz von Gleichungen, der Times-Roman-Zeichen für alle Punktgrößen beschreibt. Der Hostcomputer oder der Drucker wendet die Gleichungen an und skaliert die Gleichungen für die ausgewählte Punktgröße, so daß nicht mehrere Sätze von Bitmapdateien erforderlich sind. Die vorliegende Erfindung arbeitet entweder mit in einem ROM gespeicherten Fonts, mit Soft-Fonts oder mit der Font-Skalier-Technologie.

Ein „Glyphensatz" ist einem Soft-Font darin ähnlich, daß er vordefinierte und im Hostcomputer gespeicherte Zeichen umfaßt. Er unterscheidet sich jedoch dadurch von einem Fontsatz, daß er nicht notwendigerweise ein vollständiger Fontsatz ist und verschiedene durch den Benutzer definierte Zeichen, Grafiksymbole oder Kombinationen von verschiedenen Schrifttypen aus verschiedenen Fontsätzen umfassen kann. Ein Glyphensatz kann zum Beispiel eine Gleichung sein, die Nummern und mathematische Symbole aus verschiedenen Fontsätzen, ebenso wie einige benutzerdefinierte Grafiksymbole enthält. Ein bestimmter Glyphensatz kann groß genug sein, um einen vollständigen Zeichensatz zu enthalten, er kann aber auch nur ein einziges Zeichen enthalten. Ein weiteres Beispiel für eine Glyphensatz ist ein Formular, wie etwa eine Steuererklärung oder ein Dateneingabeformular, das in einem Dokument verwendet wird. Die vorliegende Erfindung kann das Formular erzeugen und als Glyphensatz speichern. Wenn das Formular wieder in einem Dokument verwendet wird, steht das gesamte Formular als Glyphensatz zur Verfügung. Dabei sollte beachtet werden, daß ein Glyphensatz nicht die einzige Möglichkeit darstellt, ein Formular zu implementieren. Eine andere Möglichkeit zum Implementieren eines Formulars besteht zum Beispiel darin, eine Wiedergabeelementliste RPL (= render primitive list) zu verwenden, um das Formular darzustellen. Die RPL kann Glyphensätze für Teile des Formulars verwenden, wobei die RPL alles zusammensetzt.

Einige Systeme des Standes der Technik verwenden Glyphensätze in einem begrenzten Umfang. Die Hostcomputer des Standes der Technik können eine Teilmenge eines Zeichensatzes zusammenstellen, um diese in den Drucker zu laden. Wenn ein neues Zeichen benötigt wird, kann der Hostcomputer des Standes der Technik inkrementell nur das benötigte neue Zeichen laden und zu dem bereits geladenen Glyphensatz hinzufügen. Die Systeme des Standes der Technik löschen die Glyphensätze allgemein zu Beginn einer neuen Seite, ohne Rücksicht darauf, ob der Gylphensatz in der Zukunft benötigt wird. Dadurch ist der Hostcomputer des Standes der Technik gezwungen, neue Glyphensätze zu erstellen, wenn diese auf der neuen Seite benötigt werden. Der neu erstellte Glyphensatz muß nicht mit dem vorhergehenden Glyphensatz identisch sein, wobei die periodische Rekonstruktion und das Laden der Glyphensätze zusätzliche Zeit im Druckvorgang verbraucht. Außerdem erfordern die Systeme des Standes der Technik eine große Datenmenge zusätzlich zu den Glyphensätzen, um anzugeben, welche Zeichen im Glyphensatz vorgesehen sind.

Im Gegensatz dazu, erstellt die vorliegende Erfindung einen Glyphensatz aus den verfügbaren Ressourcen und handhabt den Glyphensatz als eine einzelne Ressource. Der in der vorliegenden Erfindung verwendete Begriff „Glyphensatz" bezeichnet eine Ressource, die Teile von anderen Ressourcen wie Fonts umfaßt. Der Einfachheit halber werden die Glyphensätze als Ressourcen bezeichnet. Die vorliegende Erfindung stellt den Glyphensatz zusammen, bis er eine vorbestimmte Größe erreicht, wobei aber der Glyphensatz nicht unmittelbar zum Drucker übertragen werden darf. Der zusammengestellte Glyphensatz wird als eine Einheit behandelt, die bei Bedarf zum Drucker übertragen wird. Der Glyphensatz wird aktiv auf der Basis des zukünftigen Bedarfs des Glyphensatzes und des verfügbaren Platzes im Druckerressourcenspeicher gehandhabt, was im Gegensatz zu dem Ansatz des Standes der Technik zu sehen ist, bei dem der Druckerspeicher beim Start einer neuen Seite gelöscht wird. Der Glyphensatz der vorliegenden Erfindung enthält auch einen Header als „Inhaltsverzeichnis", das angibt, welche Zeichen im Glyphensatz vorgesehen sind, wobei der Header eine wesentlich kleinere Größe

aufweist als die Header des Standes der Technik, weil sich der Glyphensatz nicht verändert, wenn er einmal erstellt ist.

Eine „Punkttabelle" ist eine Tabelle aus Koordinaten, die verwendet wird, um ein Grafikobjekt zu definieren. Zum Beispiel kann ein Grafikobjekt wie ein Rechteck durch die Koordinaten der vier Ecken definiert werden. Entsprechend wird eine kubische Bézier-Kurve durch vier Festpunkte definiert. Um bei der Wiedergabe einer Bézier-Kurve eine fließendere Kurve zu drucken, wird die Kurvenwiedergabe häufig unter Verwendung einer hohen Auflösung durchgeführt, die die tatsächliche Auflösung des Druckers übersteigt. Die für die Wiedergabe der Kurve berechneten Linien werden geteilt, wenn das Objekt tatsächlich auf dem Drucker gedruckt wird, so daß ein kontinuierlicher wirkendes Bild erzeugt wird. Wenn die Berechnungen mit höherer Auflösung durchgeführt werden, kann die Punkttabelle die Koordinaten aller Liniensegmente enthalten, die verwendet werden, um die Bézier-Kurve wiederzugeben. Die Punkttabelle kann auch durch den Benutzer in einem Anwendungsprogramm erstellt werden, indem eine Maus oder eine andere Zeigeeinrichtung verwendet wird, um zu zeichnen, Koordinaten einzugeben oder eine Digitalisierungstabelle zu verwenden.

Ein „Brush" ist ein Grafikmuster, das typischerweise verwendet wird, um das Innere eines Grafikobjekts, etwa eines Rechtecks oder eines Kreises zu füllen. Ein Brush ist das kleinste sich wiederholende Muster, das wiederholt wird, um das gesamte Innere des Grafikobjekts zu füllen. Wenn zum Beispiel ein Objekt wie ein Kreis erstellt wird, weist das grafische Zeichenelement den Drucker an, den Kreis zu erzeugen und das Innere mit einem bestimmten Grafikmuster zu füllen. Ein Kreuzschraffierungsmuster kann zum Beispiel eine Reihe von kleinen „x"-Formen umfassen, die wiederholt werden können, um das gesamte Objekt zu füllen. In dem System der vorliegenden Erfindung werden häufig verwendete Brushes im Drucker gespeichert und verschiedene Brushes durch den Hostcomputer erzeugt.

Die Daten, die die gedruckte Seite beschreiben, werden ebenfalls als Ressource betrachtet. Der Hostcomputer enthält eine Beschreibung der Seite, die

durch ein Anwendungsprogramm wie ein Textverarbeitungsprogramm, ein Tabellenkalkulationsprogramm, ein Datenbankprogramm oder ähnliches erzeugt worden sein kann. Die vorliegende Erfindung übersetzt die Seitenbeschreibung in einen Satz von Zeichenelementen und setzt die Zeichenelemente mit den anderen Ressourcen in Beziehung, die für das Drucken des Dokuments erforderlich sind. Die Details des Übersetzungsverfahrens werden unten beschrieben.

Wie beispielhaft in dem Funktionsblockdiagramm von Fig. 2 dargestellt, ist die vorliegende Erfindung in einem Computer-Drucker-System 200 angewendet. Wie im Stand der Technik führt ein Hostcomputer 202 ein Anwendungsprogramm 204 aus, das ein Dokument enthält, das gedruckt werden soll. Wie zuvor beschrieben, werden die Ressourcen in verschiedenen Bereichen des Hostcomputers 202 gespeichert, etwa im Hostcomputerspeicher 212, der einen Festplattenspeicher enthalten kann. Die verschiedenen Speicherbereiche werden allgemein als Ressourcenspeicherbereich 206 angegeben. Der Hostcomputer 202 enthält praktisch alle Ressourcen, die für das Drucken der Dokumente verfügbar sind. Einige in ROMs gespeicherte Fonts und häufig gebrauchte Ressourcen können während des Druckauftrags im Drucker gespeichert werden. Ein Ressourcenassembler 208 untersucht das Dokument, um zu bestimmen, welche Ressourcen für das Drucken des Dokuments erforderlich sind. Wenn der Ressourcenassembler 208 das Dokument untersucht, wählt er die Ressourcen aus, die erforderlich sind, um das Dokument zu drucken, und übersetzt das Dokument in einen Satz von Zeichenelementen, der die gedruckten Seiten beschreibt. Die ausgewählten Ressourcen und Zeichenelemente sind in einem Hostressourcenspeicher 210 gespeichert. Der Hostressourcenspeicher 210 kann ein Teil des Hostcomputerspeichers 212 oder einer anderen geeigneten Speicherstelle sein. Der Ressourcenassembler 208 definiert Abhängigkeiten zwischen einem Dokument und einer Teilmenge der Ressourcen, die für das Drucken eines bestimmten Dokuments erforderlich sind. Der Ressourcenassembler 208 teilt die Informationen bezüglich der Abhängigkeiten dem Drucker 218 mit, der mit dem Hostcomputer 202 verbunden ist. Der Ressourcenassembler 208 kann vom Drucker 218 auch Information bezüglich der effektivsten Abfolge für das Drucken des

Dokuments sowie Statusinformation bezüglich der gegenwärtig im Drucker 218 vorgesehenen Ressourcen empfangen.

Der Drucker 218 enthält einen Druckerressourcenspeicher 220, der die Fähigkeit aufweist, eine begrenzte Anzahl von Ressourcen zu speichern, die aus dem Hostressourcenspeicher 210 geladen werden. Der Druckerressourcenspeicher 220 kann ein Teil eines Druckerspeichers 222 oder einer anderen Speicherstelle sein. Ein in Fig. 2 als Teil des Hostcomputers 202 gezeigter Ressourcenlader 214 verwendet die durch den Ressourcenassembler erzeugten Abhängigkeiten, um die Reihenfolge zu bestimmen, in der Ressourcen einschließlich der Zeichenelemente zum Druckerressourcenspeicher 220 übertragen werden. Der Ressourcenlader 214 bestimmt auch die Reihenfolge, in der die Ressourcen aus dem Druckerressourcenspeicher 220 freigegeben werden können oder müssen, um Platz für neue Ressourcen zu machen. Die durch den Ressourcenlader 214 übertragenen Zeichenelemente weisen den Drucker 218 an, bestimmte Ressourcen zu nützen, um ein Grafiksymbol zu erzeugen, ein Grafikobjekt zu zeichnen, eine alphanumerische Zeichen zu drucken oder ähnliches.

Ein innerhalb des Druckers 218 gezeigter Ressourcenplaner 216 kann alternativ dazu im Hostcomputer 202 lokalisiert sein. Der Ressourcenplaner 216 steuert den Zeitablauf des Druckvorgangs und den tatsächlichen Zeitablauf der Ressourcenübertragung. Der Ressourcenplaner 216 steuert auch den Zeitablauf der Löschung von Ressourcen aus dem Druckerressourcenspeicher 220 und der Anforderungen für die Übertragung bestimmter Ressourcen aus dem Hostressourcenspeicher 210. Wenn alle für eine bestimmte Seite des Dokuments erforderlichen Ressourcen im Druckerressourcenspeicher 220 sind, erzeugt der Ressourcenplaner 216 ein Ausführungssignal, um anzugeben, daß die Teilmenge der erforderlichen Ressourcen für das Drucken der aktuellen Seite verfügbar ist. Ein Ressourcenausführer 224 folgt bei Empfang des Ausführungssignals aus dem Ressourcenplaner 216 den Befehlen der Zeichenelemente und verwendet die Ressourcen aus dem Druckerressourcenspeicher 220, um eine Bitmapdatei der gegenwärtig verarbeiteten Dokumentseite zu erzeugen. Der Ressourcenausführer 224 überträgt die Bitmapdatei zu einem

Druckermechanismus 226, der wiederum veranlaßt, daß die Dokumentseite gedruckt wird.

Dabei ist zu beachten, daß die physikalische Lokation von vielen der oben beschriebenen Ressourcenblöcken nicht ausschlaggebend für den Betrieb der vorliegenden Erfindung ist. Wenn der Drucker 218 in einem Computer-Drucker-System 200 ein Laserdrucker mit einer großen Berechnungsleistung ist, können alle oben beschriebenen Ressourcenblöcke im Drucker lokalisiert sein, wobei immer noch die Aspekte der vorliegenden Erfindung genutzt werden können. Der Ressourcenplaner 216 kann zum Beispiel im Hostcomputer 202 oder im Drucker 218 lokalisiert sein, wie oben bemerkt wurde. Entsprechend kann der Druckerressourcenspeicher 220 alternativ dazu im Hostcomputer 202 lokalisiert sein. Wenn der Hostcomputer 202 in einer Umgebung wie zum Beispiel Windows™ betrieben wird, kann der Druckerressourcenspeicher 220 Teil der Ausspulfunktion sein, die im Hintergrund arbeitet, während das Anwendungsprogramm im Vordergrund arbeitet. Die Prinzipien der vorliegenden Erfindung sind auch anwendbar, weil der Druckerressourcenspeicher 220 weiterhin in seiner Größe begrenzt ist und in gleicher Weise betrieben wird, wie das der Fall sein würde, wenn der Druckerressourcenspeicher im Drucker 218 lokalisiert wäre. Der Hintergrundbetrieb ist aus der Perspektive des Anwendungsprogramms transparent. Das heißt, daß die tatsächliche Lokation des Druckerressourcenspeichers 220 nicht ausschlaggebend ist. In der Praxis kann der Hostcomputer 202 allgemein mehr Berechnungsleistung aufweisen als der Drucker 218. Deshalb werden die oben beschriebenen Ressourcenblöcke je nach der jeweiligen Berechnungsleistung und der Verfügbarkeit eines bidirektionalen Kommunikationskanals zwischen dem Hostcomputer und dem Drucker entweder im Hostcomputer 202 oder im Drucker 218 untergebracht.

Der Hostcomputer 202 speichert die Ressourcen in verschiedenen Lokationen im Hostcomputer 202 oder im Drucker 218 (im Fall von ROM-gespeicherten Zeichenfonts). Glyphensätze werden zum Beispiel durch den Ressourcenassembler 208 zusammengesetzt und als Bitmapdateien im Hostressourcenspeicher 210 gespeichert. Das Computer-Drucker-System 200 speichert auch Punkttabellen, die

verschiedene Grafikobjekte im Hostressourcenspeicher 210 wiedergeben. Die Punkttabellen werden durch den Ressourcenassembler in den Hostressourcenspeicher 210 geladen, der die Punkttabelle zu einem Datenformat umwandelt, das durch die vorliegende Erfindung verwendet wird. In anderen Fällen können die Daten, die ein Grafikobjekt beschreiben, durch ein Anwendungsprogramm in einem anderen Format als in einer Punkttabelle gespeichert werden. Der Ressourcenassembler 208 erzeugt eine Punkttabelle in der entsprechenden Form und speichert die erzeugte Punkttabelle im Hostressourcenspeicher 210. Im Gegensatz dazu werden Softfontsätze typischerweise als Dateien auf einer Festplatte (nicht gezeigt) gespeichert. Wenn der Ressourcenassembler 208 bestimmt, daß ein bestimmtes Softfontzeichen oder ein Brush benötigt wird, wird die Ressource in den Hostressourcenspeicher 210 geladen.

Sowohl bei dem Computer-Drucker-System 200 des Standes der Technik wie bei dem der vorliegenden Erfindung erzeugt das Anwendungsprogramm 204 eine Dokumentbeschreibung, die im Hostcomputerspeicher 212 oder in einer anderen geeigneten Speicherlokation wie einer Festplatte (nicht gezeigt) bewahrt werden kann. Das Anwendungsprogramm speichert das Dokument unter Verwendung einer Seitenbeschreibungssprache (PDL), die sich von einem Anwendungsprogramm zum nächsten ändern kann. Bei Systemen des Standes der Technik wandelt ein Assembler im Hostcomputer die PDL in einen Satz von Zeichenelementen um, die allgemein als Wiedergabeelemente RP (= Render Primitives) bezeichnet werden können. Die RPs umfassen alphanumerische Zeichen, Grafikobjekte oder Kombinationen aus diesen. Bei einigen Systemen des Standes der Technik übersetzt der Hostcomputer die RPs in eine Bitmapdatei der Dokumentseite mit Hilfe eines Verfahrens, das als Wiedergabe der Elementliste bezeichnet wird. Dabei sind es Bitmapdaten, die durch die Hostcomputer des Standes der Technik zum Drucker übertragen werden. Andere Hostcomputer des Standes der Technik wandeln die RPs zu eine Zwischenebenensprache wie PostScript™ oder PCL™ um.

Einige Systeme des Standes der Technik weisen tatsächlich einen Teil des Systems auf, der ähnlich wie der Ressourcenassembler arbeitet. Der Assembler des

Standes der Technik ist im Hostcomputer vorgesehen und wandelt die PDL in RPLs um. Der oben beschriebene Parser dient im Stand der Technik als zweiter Ressourcenassembler, wobei er die RPLs empfängt und die Zwischendatenstrukturen erzeugt, die erforderlich sind, um die Zwischenebenensprache zu einer entsprechenden Bitmap zu übersetzen. Der Parser dient dazu, den Code zu verarbeiten und ist nicht insbesondere dafür ausgebildet, eine gedruckte Seite zu erzeugen.

Im Gegensatz dazu verwendet das Computer-Drucker-System 200 der vorliegenden Erfindung nur einen einzigen Ressourcenassembler 208, der typischerweise im Hostcomputer 202 lokalisiert ist. Der Ressourcenassembler 208 beschäftigt sich nur mit dem Erzeugen einer gedruckten Seite, wobei der durch den Ressourcenassembler erzeugte Code dafür ausgebildet ist, Dokumente in effektiver Weise zu drucken. Der Ressourcenassembler 208 untersucht das Dokument und wandelt die PDL zu RPLs um, wobei er bestimmt, welche Ressourcen für das Drucken des Dokuments erforderlich sind. Der Ressourcenassembler 208 sammelt die ausgewählten Ressourcen und plaziert sie im Hostressourcenspeicher 210 zusammen mit den entsprechenden RPLs. Bei der vorliegenden Erfindung ist es nicht erforderlich, die Ressourcen und RPLs im Hostressourcenspeicher in einem bestimmten Format zu speichern, das die Ressourcen mit bestimmten RPLs assoziiert. Die tatsächliche Datenstruktur und das Format sind für die Verwendung in der vorliegenden Erfindung nicht wichtig. Viele verschiedene, dem Fachmann oder Laien bekannte Formate sind für den richtigen Betrieb der vorliegenden Erfindung annehmbar. Es ist lediglich eine Liste erforderlich, die die Abhängigkeiten und die Lokation der Ressourcen und RPLs angibt. Die Liste kann die Form einer Reihe von Zeigern haben, die die Lokationen angeben, an denen die Ressourcen und die entsprechenden RPLs gespeichert sind. Die Liste kann sogar in einer vorbestimmte Ausführungsabfolge von RPLs vorgesehen sein, wie weiter unten erläutert wird.

Wenn Ressourcen im Hostressourcenspeicher 210 gespeichert sind, werden sie als unbegrenzt betrachtet, da keine Beschränkung bezüglich der Größe der Datei vorgegeben ist, die die Ressourcen und die RPLs enthält, und da keine Beschränkungen bezüglich der Abfolge vorgegeben sind, in der die Ressourcen und die RPLs

gespeichert werden. Ein Dokument kann zum Beispiel durch den Benutzer erzeugt werden und dann editiert werden, um ein Grafikdiagramm nahe des Beginns des Dokuments zu enthalten. Das Anwendungsprogramm erzeugt dabei nicht die gesamte Dokumentdatei neu, um das Grafikdiagramm einzufügen. Statt dessen plaziert das Anwendungsprogramm das Grafikdiagramm am Ende der Dokumentdatei und fügt einen Zeiger an dem Punkt im Dokument ein, an dem das Grafikdiagramm eingesetzt werden soll. Der Zeiger zeigt auf die Lokation des Grafikdiagramms. Diese allgemeine Technik verwendet ein Rückwärtszeigen, d.h. der Einfügepunkt im Dokument zeigt nach hinten zu einer späteren Position in der Dokumentdatei, bei der das Grafikdiagramm gespeichert ist. Diese Technik ist auf grafische Weise in Fig. 3A dargestellt, wobei ein Dokument 300 N Seiten umfaßt. Die Seite zwei des Dokuments, die durch das Bezugszeichen 302 angegeben ist, benötigt den Font1 304, während die Seite drei 306 des Dokuments 300 das Grafikdiagramm benötigt, das durch eine Bitmap 308 wiedergegeben ist. Dabei ist zu beachten, daß der Font1 304 und die Bitmap 308 jeweils nach den Lokationen auf der Seite 302 und der Seite 306 im Dokument 300, bei denen der Font und die Bitmap benötigt werden, gespeichert sind. Die Datenzeiger 310 und 312 zeigen jeweils auf die Lokationen 304 und 308, bei denen die Ressourcen benötigt werden. Da das Dokument unbegrenzt ist, kann der Computer auf das gesamte Dokument zugreifen und kann die Zeiger in der oben beschriebenen Weise verwenden.

Beim Drucken des Dokuments hat der Computer jedoch keinen Zugriff auf die gesamte Datei. Deshalb wird das Dokument als begrenzt betrachtet, sowohl durch die Größenbeschränkung des Druckerressourcenspeichers 220 wie durch die Abfolge, in der das Dokument gespeichert ist. Die erforderlichen Ressourcen müssen im Drucker vorhanden sein, bevor sie tatsächlich benötigt werden, da ansonsten der Drucker aufgehalten wird oder überhaupt nicht in der Lage ist, eine bestimmte Seite zu drucken. Das Dokument muß unbegrenzt assembliert werden, wie in Fig. 3B angegeben ist, wo dasselbe Dokument 300 als ein begrenztes Dokument wiedergegeben ist. Der benötigte Font 316 und die Bitmap 318 erscheinen in dem Dokument 300, bevor sie tatsächlich jeweils für die Seite zwei 320 und die Seite drei 322 benötigt werden. Die Zeiger 324 und 326 zeigen jeweils

auf die Lokationen 316 und 318, wo die Ressourcen gespeichert sind. Die Zeiger 324 und 326 zeigen also vorwärts auf eine Lokation im Dokument 300, wo die Ressourcen gespeichert sind. Auf diese Weise sind die Ressourcen immer präsent, bevor sie für das Drucken des Dokuments benötigt werden. Dabei ist zu beachten, daß es nicht notwendig ist, daß die Ressourcen am Anfang des Dokuments lokalisiert sind. Es ist lediglich erforderlich, daß die Ressourcen vor der Stelle im Dokument lokalisiert sind, wo sie benötigt werden. Das Dokument 300 ist zum Beispiel alternativ in Fig. 3C als begrenztes Dokument gezeigt, wobei der Font1 320 genau vor der Lokation 320 lokalisiert ist, wo die Ressource benötigt wird. Der Zeiger 324 gibt die Lokation der benötigten Ressource an. Entsprechend ist die Bitmap 318 genau vor der Lokation 322 lokalisiert, wo sie benötigt wird, wobei der Zeiger 326 die Lokation angibt, bei der die Ressource benötigt wird.

Obwohl der Hostcomputer 202 im allgemeinen über mehr Speicher verfügt als der Drucker 218, sollte beachtet werden, daß eine Obergrenze für die Menge des Hostcomputerspeichers 212 gegeben ist, die dem Hostressourcenspeicher 210 zugewiesen werden kann. Deshalb enthält der Hostressourcenspeicher 210 nicht alle möglichen Ressourcen, die im Computer gespeichert sind. Der Hostressourcenspeicher enthält nur die Ressourcen, die für das Drucken des bestimmten Dokuments erforderlich sind, sowie die RPLs, die das Dokument beschreiben. Wenn ein bestimmter Teil des Dokuments gedruckt wurde, wird die Ressource, die für den bestimmten Teil des Dokuments erforderlich war, aus dem Hostressourcenspeicher 210 gelöscht. Einige Ressourcen können nur einmal in einem Dokument benötigt werden und werden unmittelbar nach dem Drucken dieses Teils des Dokuments gelöscht. Andere Ressourcen, wie Glyphensätze, die häufig verwendet werden können, werden im Hostressourcenspeicher 210 gespeichert, bis sie nicht mehr für ein Dokument benötigt werden.

Da der Hostcomputer 202 typischerweise über mehr Speicher verfügt als der Drucker 218, kann der Hostcomputer einen größeren Teil des Hostcomputerspeichers 212 für die Verwendung als Hostressourcenspeicher 210 zuweisen. Der Drucker 218, der über weniger Speicher verfügt, weist einen entsprechend kleineren

Druckerressourcenspeicher 220 auf. Der Druckerressourcenspeicher 220 ist nicht ausreichend groß, um den gesamten Ressourcensatz zu speichern, den der Hostressourcenspeicher enthält. Deshalb müssen Ressourcen aus dem Hostressourcenspeicher 210 in den Druckerressourcenspeicher 220 geladen werden, wenn diese durch den Drucker 218 benötigt werden. Der Drucker 218 muß die Ressourcen im Druckerressourcenspeicher 220 auf effiziente Weise nutzen und Ressourcen löschen, die nicht mehr benötigt werden oder schnell aus dem Hostressourcenspeicher 210 neu geladen werden können. Der Hostressourcenspeicher 210 wird also nur einmal mit den Ressourcen geladen, die für das Drucken des Dokuments erforderlich sind, während die Ressourcen viele Male während des Druckens des Dokuments geladen und aus dem Druckerressourcenspeicher 220 freigegeben werden können. Um die effizienteste Verwendung der Ressourcen zu bestimmen, untersucht das Computer-Drucker-System 200 der vorliegenden Erfindung das gesamte Dokument, um zu bestimmen, wie die Ressourcen in der effizientesten Weise zugewiesen werden können.

In Fig. 2 ist der Betrieb des Computer-Drucker-Systems 200 im Detail angegeben. Der Ressourcenassembler 208 wandelt die PDL in einen Satz von RPLs um und bestimmt, welche Ressourcen für den aktuellen Druckauftrag erforderlich sind. Wenn der Druckauftrag startet, beginnt der Ressourcenassembler 208, indem er das erste Band (wenn der Drucker 218 im Bandmodus betrieben wird) oder die erste Seite (wenn der Drucker 219 im Seitenmodus betrieben wird) der Daten zu betrachten, die die zu druckende Oberfläche beschreiben. In der vorliegenden Beschreibung wird die Einheitengröße des verarbeiteten Dokuments, ob es sich um ein Band oder um eine Seite handelt, als Datenblock bezeichnet. Der Ressourcenassembler 208 wählt die Ressourcen aus dem Ressourcenspeicherbereich 206 aus, der erforderlich ist, um das Dokument zu drucken. Der Ressourcenassembler bestimmt auch die Abhängigkeiten dieser Ressourcen mit bestimmten Datenblöcken. Der Ressourcenassembler 208 kann zum Beispiel bestimmen, daß ein bestimmter Fonttyp für eine Formular auf der ersten Seite benötigt wird und daß ein anderer Fonttyp für den Rest der Seite benötigt wird. Außerdem kann ein Kurvendiagramm

auf der Seite gedruckt werden; das bestimmten Grafikressourcen benötigt, etwa eine Punkttabelle und einen Brush.

Der Ressourcenassembler 208 erzeugt eine Liste, die explizit die Abhängigkeiten und den Datenblock angibt, für die die Ressourcen erforderlich sind. Dabei sollte beachtet werden, daß die Liste nicht in der Form einer Liste vorgesehen sein muß. Wie oben genannt, kann die Liste in der Form von Zeigern auf Speicherstellen vorgesehen sein oder sogar implizit durch die Abfolge definiert sein, in der die RPLs durch den Ressourcenassembler 208 erzeugt werden. Wenn zum Beispiel eine Programmierer ein Programm schreibt, das die erste RPL erzeugt und unmittelbar ausführt, ist keine ausgedrückte Liste vorgesehen, die erzeugt und im Hostressourcenspeicher 210 gespeichert wird. Es ist jedoch eine implizierte Liste vorgesehen, die durch die Reihenfolge spezifiziert ist, in der die Tasks in der RPL erzeugt werden. Für den effizienten Betrieb der vorliegenden Erfindung ist wichtig, daß der Ressourcenassembler 208 die Abhängigkeiten bestimmt und die Ressourcenblöcke über die Abhängigkeiten informiert.

Es gibt zwei unterschiedliche Typen von Abhängigkeiten, die durch den Ressourcenassembler 208 spezifiziert werden. Der erste Typ von Abhängigkeit besteht aus Operandenabhängigkeiten, die die erforderlichen Ressourcen, wie oben beschrieben, mit einem bestimmten Datenblock in Beziehung setzen. Der zweite Typ von Abhängigkeiten besteht aus Ausführungsabhängigkeiten, die die Abfolge spezifizieren, in der die RPLs verarbeitet werden. Einige Systeme des Standes der Technik können die Ausführungsabhängigkeiten nicht erfüllen, wenn sie ein Dokument drucken. Zum Beispiel trennen einige Systeme des Standes der Technik Text und Grafik und verarbeiten die beiden jeweils unabhängig. Die gedruckte Seite kann deshalb nicht dem entsprechen, was der Benutzer auf dem Bildschirm sieht. Die Systeme des Standes der Technik bieten also nicht immer „what you see is what you get" (WYSIWIG). Im Gegensatz dazu erfüllt das Computer-Drucker-System 200 immer die Ausführungsabhängigkeiten, selbst wenn diese in der Abfolge der RPLs impliziert sind, wie oben beschrieben wurde, weil das System den gesamten Datenblock verarbeitet und den Datenblock nicht in Text- und Grafikteile teilt.

Einige Ausführungsabhängigkeiten können durch den Drucker 218 spezifiziert werden, wenn eine bidirektionelle Kommunikation zwischen dem Drucker und dem Hostcomputer 202 vorgesehen ist. Wie weiter unten ausführlicher erläutert wird, kann der Drucker die Reihenfolge spezifizieren, in der die Datenseiten oder -bänder verarbeitet werden sollen, um die Effektivität des Druckvorgangs zu maximieren. Wenn die Ausführungsfabhängigkeiten durch den Drucker 218 spezifiziert werden, entspricht der Ressourcenassembler 208 diesen Abhängigkeiten. Der Ressourcenassembler 208 kann auch seine eigenen Ausführungsabhängigkeiten erzeugen, wenn eine bestimmte Zeichnungsreihenfolge im Datenblock beibehalten werden muß. Der Drucker 218 kann zum Beispiel den Ressourcenassembler 208 anweisen, die Seite zwei eines Dokuments zuerst zu verarbeiten und die Seite zwei von unten nach oben zu verarbeiten. Dies stellt eine Ausführungsabhängigkeit dar, die durch den Drucker 218 spezifiziert wird. Wenn jedoch überlappende Grafikobjekte auf der Seite zwei vorgesehen sind, muß die Zeichnungsreihenfolge dieser Objekte spezifiziert werden, damit die Objekte in der beabsichtigten Weise auf der gedruckten Seite überlappen. Der Ressourcenassembler 208 bestimmt die Ausführungsabhängigkeiten, die die Zeichnungsreihenfolge spezifizieren. Der Ressourcenassembler 208 erzeugt also eine Liste, die sowohl die Operandenabhängigkeiten wie alle Ausführungsabhängigkeiten angeben (die durch den Ressourcenassembler 208 oder den Drucker 218 spezifiziert sind).

Wie oben erläutert, kann die Liste für einige Abhängigkeiten implizit in der Abfolge der Zeichenelemente in einem Band oder einer Seite vorgesehen sein. Das Computer-Drucker-System 200 kann zum Beispiel immer zuerst den ersten RPL ausführen, wodurch eine Ausführungsabhängigkeit erzeugt wird, die nicht explizit angegeben werden muß. Um eine maximale Effektivität zu erreichen, insbesondere bei hochentwickelten Druckern und einer bidirektionalen Kommunikation, verwendet die bevorzugte Ausführungsform des Computer-Drucker-Systems 200 keine impliziten Abhängigkeiten, da sie die Ausführung unnötigerweise auf eine weniger effiziente Abfolge der Tasks beschränkt. Die obigen Beispiele der Verwendung von impliziten Abhängigkeiten wurden lediglich gegeben, um zu zeigen, daß die allge-

meinen Prinzipien der vorliegenden Erfindung verwendet werden können, um die Gesamteffektivität der Druckvorgangs zu verbessern, ohne daß die Verwendung aller erfinderischen Aspekte des Computer-Drucker-Systems 200 verwendet werden müssen. In der vorliegenden bevorzugten Ausführungsform teil der Ressourcenassembler 208 die Abhängigkeiten explizit den anderen Komponenten des Systems mit, wie dem Ressourcenlader 214, dem Ressourcenplaner 216 und dem Drucker 218.

Wenn das Computer-Drucker-System 200 über bidirektionale Kommunikationsfähigkeiten verfügt, kann der Drucker 218 Information an den Ressourcenassembler 208 bezüglich des aktuellen Status des Druckerressourcenspeichers 220 senden. Derartige Statusinformation gibt an, welche Ressourcen schon im Druckerressourcenspeicher 220 vorgesehen sind und wieviel Platz im Druckerressourcenspeicher verfügbar ist. Außerdem teilt der Drucker dem Ressourcenassembler 208 die effizienteste Abfolge für das Drucken des Dokuments mit. Dies stellt einen wichtigen Aspekt bei großen und hochentwickelten Laserdruckern dar, die doppelseitig drucken können und die Papier aus mehreren Papierschächten bedrucken können. Bei derartigen Druckern kann es auftreten, daß sich mehr als zehn Papierblätter gleichzeitig durch den Druckmechanismus bewegen. Seiten, die auf beiden Seiten des Papiers bedruckt sind (Duplexmodus) werden auf einer Seite des Papiers von oben nach unten und auf der anderen Seite des Papiers von unten nach oben verarbeitet. Unterschiedliche Seitengrößen erfordern unterschiedliche Zeitdauern im Druckermechanismus. Bestimmte Modi, wie ein Querformatmodus, können mehr Verarbeitungszeit im Druckermechanismus als andere Modi benötigen. Im Druckermechanismus eines hochentwickelten Laserdruckers können die Seiten aneinander vorbei passieren. Dies hat zur Folge, daß die effizienteste Reihenfolge für die Verarbeitung der Seiten nicht der numerischen Reihenfolge der Seiten (d.h. Seite 1, 2, 3, ...) entspricht. Das Computer-Drucker-System 200 der vorliegenden Erfindung erlaubt es dem Drucker 218, die effizienteste Abfolge für das Drucken des Dokuments zu bestimmen und diese Information dem Ressourcenassembler 208 mitzuteilen. Bei Systemen, die nur eine unidirektionale Kommunikation aufweisen, kann der Drucker 218 keine Statusinformation mitteilen oder Abfolgeinstruktionen

drucken. Der Ressourcenassembler 208 teilt dem Drucker 218 aber weiterhin die expliziten Abhängigkeiten mit, so daß der Drucker weiß, wann er die Ressourcen aus dem Druckerressourcenspeicher 220 löschen kann. Wenn nur eine unidirektionale Kommunikation möglich ist, kennt der Ressourcenassembler 208 weiterhin den Status des Druckerressourcenspeichers 220, weil der Hostcomputer 202 den Druckerspeicher 222 in dem unidirektionalen Modus verwaltet. Der Ressourcenassembler 208 weiß also, welche Ressourcen zu Beginn eines Druckauftrags bereits im Druckerressourcenspeicher 220 sind.

Wie zuvor erläutert, wandelt der Ressourcenausführer 224 typischerweise die RPLs zu Bitmapdaten um, die durch den Druckermechanismus 226 gedruckt werden. Wenn der Druckermechanismus gestartet hat, kann er nicht mit dem Druckern der Seite stoppen, da ansonsten ein Fehler auftreten würde. Wenn der Druckermechanismus aktiviert wurde, müssen die RPLs deshalb in Echtzeit zu Bitmapdaten umgewandelt werden oder bereits zuvor zu Bitmapdaten umgewandelt worden sein. Natürlich können bestimmte Drucker, wie Punktmatrixdrucker und Tintenstrahldrucker in der Mitte einer Seite stoppen, ohne einen Fehler zu erzeugen. Der Ressourcenassembler 208 kennt den aktuellen Status des Druckerressourcenspeichers 220 und die Gesamtverarbeitungsleistung des Druckers 218, wobei er jeden Datenblock untersucht, um zu bestimmen, ob der Drucker 218 die RPL für den Datenblock in Echtzeit zu einer Bitmap umwandeln kann, wenn der Druckermechanismus 226 läuft. Wenn der Drucker die RPL für den Datenblock nicht in Echtzeit umwandeln kann, weist der Ressourcenassembler 208 den Hostcomputer 202 an, die RPL in eine Bitmap umzuwandeln und die Bitmap zum Drucker 218 zu übertragen. Wenn der Druckerspeicher 222 ausreicht, um eine Bitmapdatei für die gesamte Seite zu speichern, kann der Ressourcenassembler 208 den Drucker 218 anweisen, die RPL in eine Bitmapdatei umzuwandeln und die Bitmap im Druckerspeicher 222 zu speichern, bis der Druckermechanismus 226 aktiviert ist. Die Entscheidung darüber, welcher Teil der Computer-Drucker-Systems 200 die RPL in eine Bitmap umwandelt, hängt von der relativen Komplexität der Umwandlungstask und der relativen Verarbeitungsleistung der Prozessoren in jedem Teil des Systems ab. In der bevorzugten Ausführungsform wägt der Ressourcenassembler 208 drei

Faktoren ab, wenn er bestimmt, welcher Teil des Computer-Drucker-Systems 200 die Daten verarbeitet. Diese Faktoren sind:

1. Die Zeitdauer, die der Hostcomputer 202 benötigt, um die RPL in Bitmapdaten umzuwandeln,
2. Die Zeitdauer, die der Drucker 218 benötigt, um die RPL in Bitmapdaten umzuwandeln, und
3. Die Zeitdauer, die auf dem Kommunikationskanal benötigt wird, um die RPL oder die Bitmapdaten zu übertragen.

Mit anderen Worten berechnet der Ressourcenassembler 208 die Zeitdauer, die der Hostcomputer 202 benötigt, um die RPL für einen bestimmten Datenblock zu einer Bitmapdatei umzuwandeln, und die Zeitdauer, die der Kommunikationskanal benötigt, um die Bitmapdatei zum Drucker 218 zu übertragen, und vergleicht diese mit der Zeitdauer, die der Kommunikationskanal benötigt, um die RPL zum Drucker zu übertragen, und mit der Zeitdauer, die der Drucker benötigt, um die RPL in eine Bitmapdatei umzuwandeln. Die Wahl, ob der Hostcomputer 202 oder der Drucker 218 einen Datenblock verarbeitet, wird durch eine Kostenmetrik bestimmt, die weiter unten ausführlicher erläutert wird.

Das Computer-Drucker-System 200 sorgt auch für eine Lastenbalance, indem er die Datenverarbeitung zwischen dem Hostcomputer 202 und dem Drucker 218 hin und her schiebt. Der Ressourcenassembler 208 wählt den Hostcomputer 202 oder den Drucker 218 für die Verarbeitung des Datenblocks in Abhängigkeit davon aus, welcher Teil des Systems den Datenblock am effizientesten verarbeiten kann. Wenn eine bestimmte Task zum Beispiel das Zeichnen einer großen Anzahl von Zeilen auf einer Seite erfordert und wenn der Prozessor des Hostcomputers zweimal so schnell ist, wie der Prozessor des Druckers, wird wahrscheinlich der Hostcomputer 202 angewiesen, die Daten zu verarbeiten. Wenn andererseits die Umwandlung relativ einfach ist und wenn der Drucker 218 über Speicherkapazität verfügt, um die Bitmap zu speichern, kann der Prozessor des Druckers angewiesen werden, die Daten zu verarbeiten, damit der Prozessor des Hostcomputers für die Verarbeitung des

nächsten Datenblocks frei bleibt. Dabei sollte beachtet werden, daß diese Berechnung ein dynamischer Prozeß ist, der von einem Datenblock zum nächsten variieren kann. Der Drucker 218 kann einen Datenblock verarbeiten und der Hostcomputer 2020 kann die nächsten drei Datenblöcke verarbeiten. Allgemeine Zielsetzung ist es, das Dokument auf die effizienteste Weise zu erzeugen. Das Computer-Drucker-System 200 der vorliegenden Erfindung ermöglicht dies, indem er die potentielle Berechnungsleistung von sowohl dem Hostcomputer 202 wie dem Drucker 218 nutzt.

Die Lastenbalance basiert auf verschiedenen Parametern wie der relativen Berechnungsleistung des Hostcomputers 202 und des Druckers 218, der Geschwindigkeit des Datenkommunikationskanals, den relativen Größen des Hostressourcenspeichers 210 und des Druckerressourcenspeichers 220, der Komplexität des Druckauftrags und den gegenwärtig durch den Hostcomputer 202 und den Drucker 218 durchgeführten Tasks. Wie zuvor genannt, ist die Lastenbalance ein dynamischer Prozeß, in dem der Ressourcenassembler 208 einige Seiten eines Dokuments dem Hostcomputer 202 und andere Seiten dem Drucker 218 zuweisen kann, um eine auf den oben genannten Parametern basierende Verarbeitung vorzusehen.

Die Lastenbalance kann die Datenverarbeitungsverantwortlichkeiten sogar innerhalb einer einzelnen Seite zwischen dem Hostcomputer 202 und dem Drucker 218 wechseln. Ein Beispiel dafür, daß verschiedene Teile des Computer-Drucker-Systems 200 dieselbe Seite verarbeiten, kann auftreten, wenn eine bestimmte Seite eines Dokuments zwei überlappende Grafikobjekte wie Kreise enthält. Der Ressourcenassembler 208 kann die PDL-Beschreibung des ersten Kreises zum Drucker 218 senden, das der Drucker gegenwärtig nicht druckt. Deshalb hat der Drucker 218 Zeit dafür, die PDL für den ersten Kreis zu übersetzten. Der Hostcomputer 202 kann die PDL für den zweiten Kreis übersetzen, weil der Hostcomputer 202 über mehr Berechnungsleistung verfügt als der Drucker 218 und weil der Drucker bereits mit dem Übersetzen des ersten Kreises beschäftigt ist. Der Ressourcenassembler 208 hat also eine Lastenbalance verwendet, um die

Datenverarbeitungsverantwortlichkeiten zwischen dem Hostcomputer und dem Drucker 218 aufzuteilen.

Wie zuvor festgestellt, erzeugt eine Ressource, die für einen bestimmten Datenblock des Dokuments benötigt wird, eine Abhängigkeit im Computer-Drucker-System 200 für eine bestimmten Ressource für diesen bestimmten Datenblock. Diese Abhängigkeiten können von einem Datenblock zum nächsten variieren. Der Ressourcenassembler 208 gibt die Abhängigkeiten explizit an, so daß der Drucker 218 weiß, welche Ressourcen für einen bestimmten Datenblock benötigt werden. Der Drucker weist also eine Art von Ressourcen-„Menü" auf, das die benötigten Ressourcen für jeden Datenblock angibt. Bei einer bidirektionalen Kommunikation verwaltet der Drucker 218 seinen eigenen Speicher, da die expliziten Abhängigkeiten zwischen den Datenblöcken und den Ressourcen vorgesehen wurden. Der Drucker 218 verwendet das Menü der expliziten Abhängigkeiten, um Ressourcen aus dem Hostressourcenspeicher 210 derart anzufordern, daß die Effektivität des Druckerressourcenspeichers 220 maximiert ist. Die expliziten Abhängigkeiten können zum Beispiel angeben, daß ein Datenblock einen bestimmten Fontsatz und einen bestimmten Glyphensatz benötigt, während der nächste Datenblock denselben Fontsatz aber einen anderen Glyphensatz benötigt. Der Drucker 218 kann in der Lage sein, gleichzeitig alle drei Ressourcen (den Fontsatz und die zwei Glyphensätze im Druckerressourcenspeicher 220 zu speichern. Deshalb fordert der Drucker 218 alle drei Ressourcen an.

Einen schwierigeren Aspekt der Ressourcenverwaltung stellt die Entscheidung dar, welche Ressource aus dem Druckerressourcenspeicher 220 gelöscht werden sollte. Wenn für das Drucken eines bestimmten Datenblocks eine Ressource benötigt wird, die so groß ist, daß andere Ressourcen aus dem Druckerressourcenspeicher 220 gelöscht werden müssen, kann der Drucker 218 entscheiden, welche Ressource oder welche Ressourcen aus dem Druckerressourcenspeicher 220 gelöscht werden sollen und wann die Ressourcen wieder aus dem Hostcomputer 202 für spätere Datenblöcke angefordert werden sollen. Wenn weiterhin eine Fehlerkorrektur erforderlich ist, weiß der Drucker 218,

welche Ressourcen erforderlich sind, um die verlorenen Seiten wiederherzustellen, und kann die nötigen Ressourcen aus dem Hostcomputer 202 anfordern, wenn die nötigen Ressourcen bereits aus dem Druckerressourcenspeicher 220 gelöscht wurden.

Die vorstehenden Erläuterungen sind anwendbar, wenn eine bidirektionale Kommunikation zwischen dem Hostcomputer 202 und dem Drucker 218 vorgesehen ist. Wenn nur eine unidirektionale Kommunikation erforderlich ist, wird der Druckerspeicher 222 durch den Hostcomputer 202 verwaltet. In diesem Fall bestimmt der Hostcomputer die Reihenfolge, in der die Ressourcen in den Druckerressourcenspeicher 220 geladen oder aus demselben gelöscht werden, und wann die Ressourcen geladen bzw. gelöscht werden. Obwohl der Drucker 218 seinen eigenen Speicher nicht mit einer unidirektionalen Kommunikation verwalten kann, bietet die vorliegende Erfindung wegen der Beseitigung des Parsers und der Hinzufügung von Teilen des Computer-Druckersystems, wie dem Ressourcenassembler 208 und den Host- und Druckerressourcenspeicher 210 und 220 trotzdem eine verbesserte Leistung gegenüber dem Stand der Technik. Bei dem Computer-Drucker-System 200 der vorliegenden Erfindung können die Ressourcen mehrere Male während des Druckens des gesamten Dokuments in den Druckerressourcenspeicher geladen und aus demselben gelöscht werden. Die Aufgabe, zu bestimmen, welche Ressourcen im Druckerressourcenspeicher 220 sein sollten, wird durch den Ressourcenlader 214 vorgenommen und unten ausführlicher beschrieben.

Der Ressourcenassembler 208 untersucht das Dokument mehrere Datenblöcke vor dem Ressourcenlader 214, um Ressourcen für anstehende Datenblöcke zu erzeugen. Das gestattet dem Ressourcenlader 214 vorauszublicken und die effizienteste Zuweisung von Ressourcen zu bestimmen. Einige Ressourcen können in vielen Datenblöcken de Dokuments verwendet werden und deshalb Abhängigkeiten über das Dokument aufweisen. Es kann effizienter sein, diese Ressourcen während des gesamten Druckvorgangs im Drucker 218 zu behalten, was vom verfügbaren Platz im Druckerressourcenspeicher 220 abhängt. Eine zweite Ressource kann zum Beispiel nur in der Mitte eines Dokuments benötigt werden. In

diesem Fall ist es möglich, die Ressource nicht zu laden, bis später eine andere Ressource nicht mehr durch den Drucker 218 benötigt wird, so daß mehr Speicher im Druckerressourcenspeicher verfügbar ist. Nachdem die zweite Ressource ein einziges Mal verwendet wurde, kann sie wieder aus dem Druckerressourcenspeicher gelöscht werden, um Platz für andere Ressourcen zu machen.

Die Entscheidung, wie weit vorausgeschaut wird, ist ein dynamischer Prozeß. Beim Start eines Dokuments zum Beispiel ist es das Ziel, den Druckermechanismus 226 zu starten. Deshalb wird der Ressourcenassembler 208 nur begrenzt vorausschauen, damit die Ressourcen so schnell wie möglich zum Drucker übertragen werden. Wenn der Drucker 218 jedoch die ersten Datenblocks verarbeitet, kann der Ressourcenassembler auf zukünftige Datenblocks vorausschauen, die Ressourcen für den Hostressourcenspeicher 210 auswählen und RPLs für zukünftige Seiten erstellen. Idealerweise kann der Ressourcenassembler 208 vorausschauen, um das gesamte Dokument zu untersuchen, bevor mit dem Drucken begonnen wird. Der Wunsch, den Druckermechanismus 226 zu starten, beschränkt jedoch die anfänglichen Fähigkeiten bezüglich des Vorausschauens. Es gibt eine praktische Obergrenze bezüglich des Ausmaßes der Fähigkeiten zum Vorausschauen, über die ein System verfügen sollte. Der Wunsch, die Verwendung des Hostcomputerspeichers 212 zu minimieren, so daß andere Anwendungsprogramme laufen können, beschränkt auch die Fähigkeiten des Ressourcenassemblers 208 zum Vorausschauen. Es ist das Ziel, den Druckermechanismus 226 so effizient wie möglich laufen zu lassen. Die tatsächliche Anzahl der Seiten, auf die der Ressourcenassembler 208 vorausschaut, hängt von solchen Faktoren wie der Gesamtlänge des Dokuments, der gegenwärtig durch den Druckermechanismus 226 verarbeiteten Seite des Dokuments und der Komplexität des Dokuments ab. Die Vorausschaufähigkeiten des Ressourcenassemblers 208 erhöhen die Fähigkeiten des Ressourcenladers 214 zum Steuern des Flusses der Ressourcen zum Druckerressourcenspeicher 220.

Als ein Beispiel des Betriebs des Ressourcenassemblers 208 soll der Fall betrachtet werden, daß eine besondere Seite eines Texts Teile von fünf verschiedenen Fontsätzen und eine Punkttabelle (um eine Bézier-Kurve zu zeichnen) be-

nötigt, um die Seite zu drucken. Der Ressourcenassembler 208 untersucht die Seite und erstellt eine Liste der expliziten Abhängigkeiten. Der Ressourcenassembler 208 teilt die Abhängigkeiten den anderen Teilen des Computer-Drucker-Systems 200 wie oben beschrieben mit. Gleichzeitig beginnt der Ressourcenassembler 208 auch damit, den Hostressourcenspeicher 210 zu assemblieren, der die erforderlichen Ressourcen und die RPLs, die die Seite beschreiben, enthält. Dabei ist zu beachten, daß der Ressourcenassembler 208 im bidirektionalen Modus Information bezüglich der Abfolge, in der die Datenblöcke verarbeitet werden, vom Drucker 218 empfängt. Der Einfachheit halber wird angenommen, daß der Ressourcenassembler 208 die Datenblöcke für die Seite von oben nach unten verarbeitet. Es ist eine einzige RPL vorgesehen, wenn der Drucker 218 im Seitenmodus betrieben wird, während jeweils eine andere RPL für jedes Band vorgesehen ist, wenn der Drucker im Bandmodus betrieben wird. Die RPL beschreibt den Datenblock (Seite oder Band) in einem Format, das dem Drucker 218 anweist, ein bestimmte Abfolge von Zeichen an einem bestimmten Punkt auf der Seite zu drucken. Das Computer-Drucker-System 200 verwendet diese Information, um eine Beschreibung der Zeichenfolge zu erstellen und um die Beschreibung im Hostressourcenspeicher 210 zu speichern. Der Begriff „Erstellung einer Beschreibung" weist einen Bedeutungsbereich vom Laden einer Bitmap der Zeichenabfolge von einer Speicherstelle im Hostcomputer 202 bis zum Verwenden einer Font-Skalier-Technologie, um die Bitmap der Zeichenabfolge aus einem Satz von Gleichungen zu erstellen, auf. Wenn andererseits nur eine beschränkte Anzahl von Zeichen erforderlich ist, kann der Ressourcenassembler 208 einen Glyphensatz öffnen, um nur die erforderlichen Zeichen zu speichern.

In dem vorliegenden Beispiel kann der erste Fontsatz vollständig übertragen werden. Die lediglich erforderlichen Zeichen aus dem zweiten Fontsatz können Zahlen und mathematische Symbole für eine Gleichung sein. Der Ressourcenassembler öffnet einen Glyphensatz, um die Zeichen für die Gleichung zu speichern. Der Glyphensatz kann offen bleiben, da der nächste Teil der Seite eine begrenzte Anzahl von kursiven Zeichen benötigt (dritter Font). Dabei sollte beachtet werden, daß die Größe eines Glyphensatzes dynamisch variabel ist. Zum Beispiel bei Beginn des Druckvorgangs ist es die Zielsetzung, daß der Druckermechanismus 226 so

schnell wie möglich zu arbeiten beginnt. Um diese Zielsetzung zu erreichen, kann der Ressourcenassembler 208 kleine Glyphensätze für die ersten Datenblöcke des Dokuments verwenden, damit die Glyphensätze so schnell wie möglich zum Druckerressourcenspeicher 220 übertragen werden können. Das gibt dem Druckermechanismus 226 etwas zu tun, während der Ressourcenassembler 208 Ressourcen für die folgenden Datenblöcke zusammenstellt. Die Größe der folgenden Glyphensätze wird allgemein durch Parameter wie die Größe des Druckerressourcenspeichers 220 und die Datenübertragungsrate zwischen dem Hostcomputer 202 und dem Drucker 218 bestimmt. Der Ressourcenassembler 208 hält den Glyphensatz offen, bis er eine vorbestimmte Größe erreicht.

Wie zuvor beschrieben, können die Glyphensätze Zeichen aus verschiedenen Fontsätzen enthalten. Umgekehrt können Zeichen aus demselben Fontsatz aufgrund der Abhängigkeiten in verschiedenen Glyphensätzen gespeichert werden. Zum Beispiel können einige der Zeichen, die in den oben genannten mathematischen Formeln verwendet werden, in einer zweiten Gleichung verwendet werden, die in einem darauffolgenden Datenblock gedruckt wird. Die zweite Gleichung kann auch zusätzliche Zeichen aus dem zweiten Fontsatz sowie Zeichen aus einem vierten und einem fünften Fontsatz verwenden. Der Ressourcenassembler 208 kann einen zweiten Glyphensatz erstellen, der nur die zusätzlichen Zeichen enthält, die für die zweite Gleichung erforderlich sind. Wenn der Ressourcenausführer 224 die RPLs und die Ressourcen zu einer Bitmapdatei verarbeitet, verwendet er die Zeichen aus beiden Glyphensätzen, um eine Bitmapdatei für die zweite Gleichung zu erstellen. Die RPL zum Plazieren eines Glyphensatzes ist in einem Format, das angibt, welcher Gylphensatz und welches Zeichen an einer bestimmten Position auf der gedruckten Seite plaziert wird. Die RPL für die zweite Gleichung des Beispiels kann die folgende Abfolge aufweisen:

Glyphensatz 1; Zeichen 1;
Glyphensatz 1; Zeichen 2;
Glyphensatz 1; Zeichen 3;
Glyphensatz 1; Zeichen 12;

Glyphensatz 2; Zeichen 1;
Glyphensatz 2; Zeichen 2;
Glyphensatz 1; Zeichen 17;
Glyphensatz 2; Zeichen 3;
Glyphensatz 2; Zeichen 4;
Glyphensatz 2; Zeichen 4;
Glyphensatz 2; Zeichen 5;
Glyphensatz 2; Zeichen 6; und
Glyphensatz 2; Zeichen 7.

Dabei ist zu beachten, daß die Verwendung von beiden Glyphensätzen in einer einzigen RPL erfordert, daß beide Glyphensätze gleichzeitig im Druckerressourcenspeicher 220 sind. Wenn der erste Glyphensatz aus der Druckerressourcenspeicher 220 gelöscht ist, bestimmt der Ressourcenlader 214, daß der erste Glyphensatz neu aus dem Hostressourcenspeicher geladen werden muß.

Der Ressourcenplaner 216 steuert die Zeitabfolge der Anforderung, so daß der Druckerressourcenspeicher 220 nicht überläuft und so daß die Ressourcen im Druckerressourcenspeicher in zeitlicher Abstimmung verfügbar sind. Die Systeme des Standes der Technik laden ganze Fonts und versuchen nicht den Druckerspeicher zu verwalten. Das kann zu Speicherüberläufen führen, bei denen der Druckauftrag nicht abgeschlossen werden kann. Selbst Systeme, die ein inkrementierendes Laden durchführen können, versuchen nicht, den Druckerspeicher zu verwalten, außer daß sie periodisch die geladenen Fonts löschen. Im Gegensatz dazu spart das Computer-Drucker-System 200 der vorliegenden Erfindung Zeit und Druckerspeicher durch das Zusammenstellen von Zeichen zu Glyphensätzen, indem nur die benötigten Zeichen zum Druckerressourcenspeicher 220 übertragen werden und der Glyphensatz wie oben beschrieben aktiv verwaltet wird. Auf diese Weise wird die Gesamteffizienz des Druckvorgangs erhöht.

Der Betrieb des Ressourcenassemblers 208 umfaßt das Bestimmen der Ressourcen-Abhängigkeiten, das Mitteilen der Information zu anderen Teilen des Computer-Drucker-Systems 200 und das Verarbeiten der Dokument-Beschreibung in der effizientesten Weise. Der Ressourcen-Assembler 208 erzeugt auch RPLs, die Datenblöcke beschreiben und speichert die RPLs und die Ressourcen im Hostressourcenspeicher 210.

Der Ressourcenlader 214 ist dafür verantwortlich, die Reihenfolge zu bestimmen, in der die Ressourcen in den Druckerressourcenspeicher 220 geladen und aus demselben freigegeben werden. Der Ressourcenlader 214 hat immer über den Ressourcenassembler 208 Zugriff auf die Systemabhängigkeiten, so daß die effizienteste Abfolge für das Laden und das erneute Laden der Ressourcen bestimmt werden kann. Der Ressourcenlader 214 kann im Hostcomputer 202 oder im Drucker 218 lokalisiert sein, was von den Kommunikationsfähigkeiten des Computer-Drucker-Systems 200 abhängt. Wenn nur eine unidirektionale Kommunikation vom Hostcomputer 202 zum Drucker 218 vorgesehen ist, ist der Ressourcenlader 214 immer im Hostcomputer 202. Der Druckerspeicher 222 wird also durch den Hostcomputer 202 verwaltet. Wenn jedoch bidirektionale Kommunikationsfähigkeiten vorgesehen sind, kann der Ressourcenlader 214 im Drucker 218 vorgesehen sein, um dem Drucker das Verwalten seines eigenen Speichers zu erlauben. Der Ressourcenlader 214 steuert die Übertragung sowohl der RPLs wie der Ressourcen zum Drucker 218.

Wie oben bemerkt, ist der Hostressourcenspeicher 210 groß genug, so daß die durch den Ressourcenassembler 208 zusammengestellten Ressourcen nur ein einziges Mal in den Hostressourcenspeicher geladen werden. Der Hostressourcenspeicher 210 kümmert sich nicht um die Größe der Ressourcen oder um die durch die Größe der Druckerressourcenspeichers 220 auferlegten Beschränkungen. Andererseits ist der Druckerressourcenspeicher 220 in seiner Größe begrenzt und sind die Ressourcen durch die Größenbegrenzung eingeschränkt. Um den Druckerressourcenspeicher 220 effizient zu verwalten, betrachtet der Ressourcenlader 214 die Größe jeder Ressource bereits im Druckerressourcenspeicher 220 und

betrachtet weiterhin die Ressourcenabhängigkeiten (zuvor durch den Ressourcenassembler 208 bestimmt), wobei er die Reihenfolge bestimmt, in der die Ressourcen in den Drucker geladen und aus demselben freigegeben werden, so daß der Druckerressourcenspeicher nicht überläuft. Der Ressourcenlader 214 kann also eine bestimmte Ressource viele Male während des Druckens eines Auftrags laden und wieder freigeben.

Dabei ist zu beachten, daß der Ressourcenlader 214 eine bestimmte Ressource freigeben kann, wenn sie nicht mehr benötigt wird. Der Drucker 218 kann die besondere Ressource nicht unmittelbar aus dem Druckerressourcenspeicher 220 löschen, da die Ressource weiterhin im Drucker 218 gebraucht werden kann. Da der Hostcomputer 202 und der Drucker 218 asynchron operieren, kann die Freigabe einer Ressource durch den Ressourcenlader 214 nicht unmittelbar das Löschen der Ressource aus dem Druckerressourcenspeicher 220 veranlassen. Die „Freigabe" und das „Löschen" einer Ressource ist dabei nicht synonym. Eine Ressource wird freigegeben, wenn der Ressourcenlader 214 bestimmt, daß eine Ressource aus dem Druckerressourcenspeicher 220 entfernt werden sollte. Aus der Perspektive des Ressourcenladers 214 ist die Ressource dann nicht mehr im Drucker 218 vorhanden. Der Ressourcenlader 214 spezifiziert dann die nächste Ressource, die geladen oder freigegeben werden soll. Eine Ressource wird gelöscht, wenn der Drucker 218 die Ressource nicht mehr im Drucker benötigt und die Ressource tatsächlich aus dem Druckerressourcenspeicher 220 löscht. Der Ressourcenlader 214 ist nur an der Größe jeder Ressource interessiert und daran, ob es hinsichtlich der Effizienz sinnvoll ist, daß eine bestimmte Ressource im Druckerressourcenspeicher 220 vorhanden ist. Der Ressourcenlader 214 verfolgt die Größe der Druckerressourcenspeichers 220 und den darin verfügbaren Platz sowie den aktuellen Status des Druckerressourcenspeichers 220 (d.h. welche Ressourcen im Druckerressourcenspeicher vorhanden sind) und bestimmt, welche Ressourcen behalten und welche freigegeben werden sollen. Der Ressourcenlader 214 betrachtet die expliziten Abhängigkeiten für sowohl die aktuellen RPLs wie die zukünftige RPLs. Dabei ist zu beachten, daß der Ressourcenlader 214 nur an der Reihenfolge interessiert ist, in der die Ressourcen geladen und freigegeben werden

sollten; er kümmert sich nicht um den tatsächlichen Zeitablauf der Ressourcenwechsel. Der Zeitablauf der Wechsel im Druckerressourcenspeicher 220 wird durch den Ressourcenplaner 216 gesteuert.

Wie oben genannt, machen es die explizit angegebenen Abhängigkeiten dem Ressourcenlader 214 leichter, die Abfolge zum Laden der Ressourcen in den Druckerressourcenspeicher 220 zu bestimmen. Die schwierigere Aufgabe besteht darin, zu bestimmen, wann die Ressourcen aus dem Druckerressourcenspeicher 220 freigegeben werden sollten, um Platz für neue Ressourcen zu machen. Es ist offensichtlich, daß eine nicht mehr verwendete Ressource bedenkenlos gelöscht werden kann. Wenn die Ressourcen jedoch später noch benötigt werden, muß der Ressourcenlader 214 entscheiden, welche Ressource freigegeben werden soll, um Platz für neue Ressourcen zu machen. Bei vielen Cache-Systemen des Standes der Technik, wird allgemein so vorgegangen, daß das Item gelöscht wird, das am längsten nicht benutzt wurde (d.h. es wird die Ressource gelöscht, deren Verwendung am längsten zurückliegt). Dieser Ansatz ist jedoch nicht von Nutzen, um vorauszusagen, welche Ressourcen in der Zukunft benötigt werden. Aufgrund der expliziten Abhängigkeiten kann das Computer-Drucker-System 200 ein hellseherisches Cache-Speichern der Ressourcen vornehmen, um das effizienteste Speichern der Ressourcen für spätere Datenblöcke des Dokuments vorauszusagen. Die Ressourcen werden auf der Basis der Reihenfolge, in der die Ressourcen verwendet werden, des Platzes, der für das Speichern einer Ressource erforderlich ist, und der Zeit, die für das erneute Laden einer Ressource erforderlich ist, wenn die Ressource aus dem Druckerressourcenspeicher 220 freigegeben werden muß, verwaltet. Der Ressourcenlader 214 verwendet die expliziten Abhängigkeiten, um eine „Zeitlinie" zu erstellen, mit der der Ressourcenlader die aktuell im Druckerressourcenspeicher 220 vorhandenen Ressourcen betrachtet und bestimmt, welche Ressource am weitesten in der Zeit entfernt verwendet wird. Wie oben genannt, betrachtet der Ressourcenlader 214 aber auch die Größe der zu löschenden Ressource und die für das spätere Neuladen der Ressource erforderliche Zeit.

Als Beispiel eines hellsehenden Cache-Speicherns soll angenommen werden, daß der Druckerressourcenspeicher 220 bereits zehn Ressourcen enthält (die allgemein für dieses Beispiel von 1 bis 10 numeriert sind), wobei der Drucker 218 eine Ressource 11 für einen bestimmten Datenblock benötigt. Der Ressourcenlader 214 betrachtet die Zeitlinie und kann bestimmen, daß zum Beispiel die Ressource 8 in der Zeit am weitesten entfernt ist. Wenn die Ressource 8 jedoch klein ist, kann der Druckerressourcenspeicher 220 nach ihrer Freigabe immer noch nicht über ausreichenden Platz für das Laden der erforderlichen Ressource 11 verfügen. Deshalb betrachtet der Ressourcenlader 214 die Zeitlinie ein zweites Mal, um die nächste Ressource zu bestimmen, die am weitesten in der Zeit entfernt verwendet wird. Es kann zum Beispiel die Ressource 2 freigegeben werden. Wenn die Freigabe der Ressource 2 jedoch mehr freien Platz im Druckerressourcenspeicher 220 erzeugt als nötig ist, und wenn das neue Laden des Ressourcenzahl 2 in der Zukunft sehr zeitaufwendig ist, kann der Ressourcenlader 214 wieder die Zeitlinie betrachten, um statt dessen eine oder mehrere andere Ressourcen freizugeben. In diesem Beispiel kann der Ressourcenlader 214 die Ressourcen 7 und 5 freigeben und nicht die Ressourcen 2 und 8, um im Druckerressourcenspeicher 220 für die benötigte Ressource 11 Platz zu machen. Diese Beschreibung dient lediglich als ein Beispiel für die verschiedenen Parameter, die der Ressourcenlader 214 betrachtet, um den Druckerressourcenspeicher 220 zu verwalten.

Während der Ressourcenlader 214 die Reihenfolge bestimmt, in der die Ressourcen in den Druckerressourcenspeicher 220 geladen werden und aus demselben freigegeben werden, wird die tatsächliche Zeitsteuerung der Ressourcenverwaltung durch den Ressourcenplaner 216 durchgeführt. Der Ressourcenplaner 216 kann als das Druckerbetriebssystem angesehen werden. Wie oben erläutert, muß der Ressourcenplaner 216 nicht physikalisch im Drucker 218 lokalisiert sein. In einem Computer-Drucker-System 200 mit unidirektionaler Kommunikation kann der Ressourcenplaner 216 im Hostcomputer 202 lokalisiert sein und den Druckerspeicher 222 aus dem Hostcomputer verwalten. Wenn das Computer-Drucker-System 200 bidirektional kommunizieren kann, ist der Ressourcenplaner im Drucker 218 vorgesehen, was dem Drucker das Verwalten seines eigenen Druckerspeichers

222 erlaubt. Da der Hostcomputer 202, der Drucker 218 und der Druckermechanismus 226 im Computer alle asynchron operieren, muß der Ressourcenplaner 216 das gesamte Timing steuern, so daß keine Konflikte zwischen den drei asynchronen Teilen auftreten. Der Ressourcenplaner 216 initiiert und steuert den gesamten Druckzeitablauf, synchronisiert den Betrieb mit dem Druckermechanismus 226 und entscheidet, wann eine bestimmte Ressource im Druckerressourcenspeicher 220 angenommen wird.

Der Ressourcenplaner 216 entscheidet auch, wann eine bestimmte Ressource aus dem Druckerressourcenspeicher 220 gelöscht werden soll. Wie zuvor beschrieben, ist es die Aufgabe des Ressourcenladers 214, die Reihenfolge für das Laden und Freigeben der Ressourcen zu spezifizieren. Der Ressourcenplaner 216 bestimmt, wann der Drucker 218 eine bestimmte Ressource nicht mehr braucht, die zuvor durch den Ressourcenlader 214 freigegeben wurde. Wie der Ressourcenlader 214 hat auch der Ressourcenplaner 216 Zugriff auf die expliziten Abhängigkeiten, die durch den Ressourcenassembler 208 erzeugt wurden. Im Gegensatz zu dem Ressourcenlader 214 ist der Ressourcenplaner 216 nur daran interessiert, ob die benötigten Ressourcen für die aktuelle Seite im Druckerressourcenspeicher 220 vorhanden sind.

Wenn alle Abhängigkeiten für die aktuelle Seite erfüllt werden (d.h. alle benötigten Ressourcen im Druckerressourcenspeicher 220 sind), erzeugt der Ressourcenplaner 216 eine Ausführungssignal, das den Druckermechanismus 226 für das Drucken der Seite aktiviert. Wie weiter unten ausführlicher erläutert wird, muß der Druckermechanismus in Echtzeit mit den Bitmapdaten versorgt werden, wenn die der Druck einer Seite aktiviert wurde; die Seite würde ansonsten nicht richtig gedruckt werden, da der Druckermechanismus nicht in der Mitte einer Seite stoppen kann. Dabei ist zu beachten, daß ein doppelseitig druckender Drucker ein Ausführungssignal für jede Seite eines Blattes benötigt (d.h. der Druckvorgang kann zwischen den beiden Seiten des Papiers stoppen). Der Ressourcenplaner 216 bestimmt, wann der Druckermechanismus aktiviert werden kann und erzeugt das

Ausführungssignal, um den Druckmechanismus für das Drucken einer Seite zu aktivieren.

Der Ressourcenplaner 216 erfüllt bei einer unidirektionalen und bei einer bidirektionalen Kommunikation ähnliche Funktionen. Bei einem unidirektional kommunizierenden System erzeugt der Ressourcenplaner 216 ein BUSY-Flag in der Hardwareschnittstelle, das dem Hostcomputer 202 den Druckerstatus angibt. Der Ressourcenplaner entscheidet auch, wann in der Zeit eine Ressource tatsächlich aus dem Druckerressourcenspeicher 218 gelöscht wird. In einem bidirektionalen Kommunikationssystem verwaltet der Ressourcenlader 214 den Druckerspeicher 222 aus dem Drucker 218 und fordert vom Hostcomputer 202 bestimmte Ressourcen an. Außerdem überwacht der Ressourcenplaner 216 den Druckvorgang und informiert den Hostcomputer 202, wann eine Seite den letzten Papierstau-Sensor im Druckermechanismus 226 passiert hat. Auf diese Weise weiß der Hostcomputer 202, daß er die mit dieser Seite assoziierten Ressourcen nicht länger für eine Fehlerkorrektur zu behalten braucht. Der Ressourcenplaner 216 kann auch den Papierpfad für einen Druckauftrag planen. Das ist insbesondere bei großen Druckern wichtig, die mehrere Papierschächte aufweisen und mit mehreren Papiergrößen und -pfaden arbeiten. Das Planen des optimalen Papierpfads verbessert die Gesamteffizienz des Druckvorgangs.

Der Ressourcenausführer 224 nimmt das Ausführsignal aus dem Ressourcenplaner 216 an und wandelt die RPLs zu einer Bitmap um, die durch den Druckermechanismus verwendet werden kann, um die Seite tatsächlich zu drucken. Andere Ressourcen können bereits im Druckerressourcenspeicher 220 in der Form einer Bitmap vorhanden sein. Der Ressourcenausführer 224 verwendet die Ressourcen, die gegenwärtig im Druckerressourcenspeicher 220 verfügbar sind, um die Bitmap zu erzeugen. Wie zuvor erläutert, operieren einige Drucker im Bandmodus. Das Computer-Drucker-System 200 der vorliegenden Erfindung arbeitet mit Druckern, die im Bandmodus oder im Seitenmodus operieren. Der Ressourcenausführer 224 ist auf einen Echtzeitbetrieb beschränkt, wenn ein Bandmodus verwendet wird. Wenn also der Druckmechanismus 226 in Echtzeit aktiviert wurde, muß der

Ressourcenausführer alle RPLs jeweils bandweise in Echtzeit zu einer Bitmap umwandeln, da ansonsten ein Fehler auftritt. Wenn der Drucker 218 im Seitenmodus operiert (im Gegensatz zum Bandmodus), ist keine Echtzeit-Aktivierung vorgesehen. Der Ressourcenausführer 224 kann die gesamte Seite zu einer Bitmap umwandeln, bevor die Bitmap an den Druckermechanismus übertragen wird. Das Computer-Drucker-System 200 der vorliegenden Erfindung kann entweder im Seitenmodus oder im Bandmodus operieren. Die tatsächliche Umwandlung einer RPL zu einer Bitmapdatei ist dem Fachmann wohlbekannt und wird hier nicht erläutert.

Der Druckermechanismus 226 nimmt die Bitmapdaten aus dem Ressourcenausführer 224 an und veranlaßt, daß die Bitmapdaten auf der Seite gedruckt werden. Die Verwendung des Druckermechanismus 226 ist dem Fachmann ebenfalls wohlbekannt und wird hier nicht erläutert.

Wenn die Bitmapdaten für eine Seite des Dokuments durch den Druckermechanismus 226 verarbeitet werden, bewegt sich das Papier durch den Drucker 218. Es ist eine Anzahl von Sensoren im Druckermechanismus 226 vorgesehen, die Fehler wie einen Papierstau oder eine niedrige Tonerkonzentration feststellen. Wenn ein Papierstau auftritt, verfügen die Systeme des Standes der Technik bereits über die Daten in der Form einer Bitmap, um die gestaute Seite neu zu drucken. Wenn das Computer-Drucker-System 200 über bidirektionale Kommunikationsfähigkeiten verfügt, werden die Bitmapdaten jedoch nicht im Drucker 218 behalten, sondern es werden Fehlerkorrekturdaten im Hostcomputer 202 erzeugt. Die Systeme des Standes der Technik können eine schnellere Fehlerkorrektur als die vorliegende Erfindung bieten, da die Bitmapdaten beim Auftreten eines Papierstaus bereits im Druckerspeicher warten, neu gedruckt zu werden. Papierstaufehler treten jedoch beim normalen Drucken so selten auf, daß es für den gesamten Druckvorgang effizienter ist, mit dem Verarbeiten der Daten für spätere Seiten fortzufahren und sich nicht um die effizienteste Technik für die Fehlerkorrektur zu kümmern. Das Computer-Drucker-System 200 der vorliegenden Erfindung beschäftigt sich mit der effizientesten Technik zum Drucken des gesamten Dokuments.

Die Systeme des Standes der Technik können die Daten für die nächste Seite nicht sofort verarbeiten, weil der Druckerspeicher gezwungen ist, die Bitmapdaten zu behalten, bis die Seite den letzten Papierstau-Sensor passiert hat. Bei einem typischen Druckermechanismus dauert es ungefähr zehn Sekunden, um ein Papierblatt zu nehmen, ein Bild auf dem Papier zu erzeugen und das Papier in einen Papierschacht fallen zu lassen. Die vorliegende Erfindung fährt damit fort, die Daten für spätere Seiten in einem Dokument zu verarbeiten, wobei erwartet wird, daß kein Papierstau auftritt. Während der Zeit, in der die Systeme des Standes der Technik darauf warten, daß die gedruckte Seite den letzten Papierstau-Sensor passiert, kann das Computer-Drucker-System 200 Ressourcen zusammenstellen, die PDL in RPLs übersetzen und den Fluß der Ressourcen im Druckerressourcenspeicher 220 für mehrere Seiten speichern.

Bei dem unwahrscheinlichen Auftreten eines Papierstaus verarbeitet der Hostcomputer 202 die Seite erneut. Es ist kein wirklicher Zeitverlust gegeben, da der Bediener beim Auftreten eines Papierstaus eingreifen muß, um auf physikalische Weise die gestaute Seite oder die gestauten Seiten zu entfernen. Während der Bediener die gestauten Seiten entfernt, bestimmt der Ressourcenlader 214, für welche Seiten eine Fehlerkorrektur erforderlich ist, und beginnt damit, die erforderlichen Ressourcen und RPLs in den Drucker 218 zu laden. Die expliziten Abhängigkeiten vereinfachen die Fehlerkorrektur, da der Ressourcenlader 214 die Liste der expliziten Abhängigkeiten betrachtet, um zu bestimmen, welche Ressource für die Fehlerkorrektur benötigt wird. Im Drucker 218 kann zum Beispiel ein Stau von Seiten mit den Seitenzahlen zwei bis fünf aufgetreten sein, wobei die Seiten zwei und drei doppelseitig und die Seiten vier und fünf einseitig gedruckt werden. Wenn der Drucker 218 zuvor die Druckreihenfolge von Seite drei (von unten nach oben), über Seite zwei (von oben nach unten), über Seite vier (von oben nach unten) zu Seite fünf (von oben nach unten) festgelegt hat, verwendet der Ressourcenlader 214 die expliziten Abhängigkeiten, um Ressourcen und RPLs in der effizientesten Weise für die Durchführung der Fehlerkorrektur anzufordern. Diese Aktivitäten können vorgenommen werden, während der Bediener das gestaute Papier entfernt. Auf diese Weise verliert das Computer-Drucker-System 200 im Vergleich zu den

Systemen des Standes der Technik keine Zeit bei der Fehlerkorrektur. Weiterhin wird die Effizienz des Druckvorgangs beträchtlich verbessert, indem angenommen wird, daß normalerweise kein Papierstau auftritt. Das Computer-Drucker-System 200 kann also ein Dokument in einer viel kürzeren Zeit verarbeiten als eines der Systeme des Standes der Technik.

Wie zuvor genannt, kann das Computer-Drucker-System 200 der vorliegenden Erfindung mit einer unidirektionalen Kommunikation vom Hostcomputer 202 zum Drucker 218 operieren, es kann aber auch mit einem vollständig bidirektionalen Kommunikationskanal zwischen dem Hostcomputer 202 und dem Drucker 218 funktionieren. Wenn die Hardware des Hostcomputers 202 oder des Druckers 218 keine bidirektionale Kommunikation unterstützen können, ist lediglich eine unidirektionale Kommunikation möglich. Auch mit den Beschränkungen einer unidirektionalen Kommunikation stellt das Computer-Drucker-System 200 der vorliegenden Erfindung eine Verbesserung gegenüber dem Stand der Technik dar. In einigen Fällen kann eine bidirektionale Kommunikation durch sowohl das Computersystem 202 wie den Drucker 218 unterstützt werden, wobei jedoch die Latenzzeit des bidirektionalen Kommunikationskanals so lang ist, daß es unmöglich ist, eine volle bidirektionale Kommunikation zu unterstützen. Dabei kann das Computer-Drucker-System 200 der vorliegenden Erfindung eine begrenzte bidirektionale Kommunikation zwischen dem Drucker 218 und dem Hostcomputer 202 unterstützen. Dieser Modus ist zwar nicht so effizient wie eine volle bidirektionale Kommunikation, ist aber einer unidirektionalen Kommunikation vorzuziehen. Eine begrenzte bidirektionale Kommunikation bietet dem Hostcomputer eine bessere Benachrichtigung bezüglich des Status bzw. bezüglich von Fehlern als eine unidirektionale Kommunikation. Die Datenverarbeitung kann wie bei einer unidirektionalen Kommunikation fortfahren, wobei das Computer-Drucker-System bei Auftreten eines Fehlers jedoch die Fehler- und Statusinformation für die Fehlerkorrektur verwenden kann.

Einige Laserdruckersysteme, die die Druckersteuersprache PCL verwenden, sehen das Einstecken einer Softwarekassette in den Drucker vor. Die Kassette kann

zusätzliche Fonts enthalten. Das Computer-Drucker-System 200 kann eine derartige Kassette verwenden, um die im Drucker vorzusehenden erforderlichen Komponenten der Erfindung für den Drucker zur Verfügung zu stellen. In einer Ausführungsform der vorliegenden Erfindung verfügt das Computer-Drucker-System über die Fähigkeit, in einem ersten Modus unter Verwendung von PCL oder in einem zweiten Modus unter Verwendung der vorliegenden Erfindung zu operieren. In dieser Ausführungsform kann das Computer-Drucker-System 200 automatisch zwischen den beiden Modi hin und her schalten. Das erlaubt es dem Computer-Drucker-System 200 mit anderen Anwendungen, wie etwa DOS-Anwendungen kompatibel zu bleiben. Durch das Hin- und Herschalten zwischen den beiden Modi, weist das Computer-Drucker-System eine größere Kompatibilität zu Systemen des Standes der Technik auf.

Ein weiterer wichtiger Aspekt des erfinderischen Systems ist die Bestimmung der Kostenmetrik. Wie zuvor erwähnt, bestimmt der Ressourcenassembler 208 (siehe Fig. 2), ob der Ressourcenausführer 224 die RPLs für einen Datenblock in Echtzeit umwandeln kann. Als weitere Verbesserung bestimmt der Ressourcenassembler 208, ob der Hostcomputer 202 oder der Drucker 218 die RPLs für einen Datenblock am effizientesten umwandeln kann. Damit der Hostcomputer 202 an der normalerweise durch den Drucker verrichteten Arbeit teilnehmen kann, muß der Hostcomputer 202 im voraus wissen, wie lange der Drucker 218 für das Verarbeiten einer RPL braucht. Die für bestimmte Druckeraktivitäten erforderliche Zeitdauer muß bei der Entscheidung berücksichtigt werden, was zum Drucker 218 gesendet wird, damit der Drucker keinen Fehler verursacht. Die vorliegende Erfindung verwendet ein als Kostenmetrik bezeichnetes statistisches Verfahren, um zu erfahren, wie lange das Drucken dauert. Die Verwendung der Kostenmetrik erlaubt es dem Hostcomputer 202 und dem Drucker 218 die Verarbeitung der Ressourcen zwischen dem Hostcomputer und dem Drucker aufzuteilen, um den Datendurchsatz zu maximieren. Der Drucken verrichtet nun nicht die gesamte Arbeit und der Hostcomputer 202 sieht in der Zeitspanne, während der der Drucker das Papier einzieht, Verarbeitungsfähigkeiten vor und ermöglicht auf diese Weise eine beträchtliche Verbesserung der Gesamtdruckzeit. Unter Kosten ist hier die Druckzeit für eine Seite oder ein Band

von Daten zu verstehen. Wegen des einzigartigen, oben beschriebenen Ansatzes für die Datenverarbeitung kann die vorliegende Erfindung Daten zum Drucker 218 geben, so daß der Drucker eine beliebige Datenseite in einer Zeitspanne von maximal dreißig Sekunden verarbeiten kann.

In einem typischen Computer-Drucker-System des Standes der Technik wird eine Zeitauslösung verwendet, um dem Benutzer anzuzeigen, daß eine Fehlerbedingung vorliegt. Die Zeitauslösung entspricht der Zeitdauer, in der erwartet wird, daß der Drucker oder ein anderes Peripheriegerät Daten verarbeitet und nicht mit dem Hostcomputer kommuniziert. Wenn der Drucker länger als die vorbestimmte Zeitauslösung nicht mit dem Hostcomputer kommuniziert, nimmt der Hostcomputer an, daß eine Fehlerbedingung vorliegt. Die Zeitauslösung steht im Stand der Technik jedoch nicht mit einem bestimmten Vorgang im Drucker in Beziehung, sondern ist eine willkürliche Zeitdauer, die derart ausgewählt wurde, daß die meisten Druckaufträge ohne das Signalisieren eines Fehlers ausgeführt werden können. Die typische Zeitauslösung beträgt im Stand der Technik ungefähr zwei Minuten. Wie oben genannt, ist die Zeitdauer von zwei Minuten willkürlich auf der Grundlage gewählt, daß ein typischer Druckauftrag nicht länger als zwei Minuten dauert. Es könnten ebensogut drei Minuten für die Zeitauslösung gewählt werden. Das System des Standes der Technik nimmt an, daß jeder Druckauftrag, der mehr als zwei Minuten dauert, ein Fehler sein kann. Es treten jedoch häufig Druckaufträge auf, die mehr als zwei Minuten benötigen. Wenn das der Fall ist, läuft die Zeitauslösung ab und der Hostcomputer erzeugt eine falsche Fehlermeldung. Mit anderen Worten erzeugt der Hostcomputer jedesmal eine Fehlermeldung, wenn die Zeitauslösung überschritten wird, auch wenn kein Fehler aufgetreten ist.

Im Gegensatz dazu erlaubt das Computer-Drucker-System 200 die genaue Bestimmung der für den Drucker erforderlichen Zeit, um eine Datenseite zu verarbeiten. Der Hostcomputer 202 kann die Zeitauslösung setzen, um die tatsächlich erwartete Verarbeitungszeit im Drucker 218 wiederzugeben. Die Zeitauslösung steht also in der vorliegenden Erfindung tatsächlich in Beziehung mit der Verarbeitungszeit. Die Verarbeitungsaufwand des Hostcomputers 202 und des

Druckers 218 wird reduziert, wenn sowohl der Hostcomputer wie der Drucker wissen, wann der andere frei kommunizieren kann.

Wenn der Hostcomputer 202 direkt mit dem Drucker 218 verbunden ist, kann der Hostcomputer 202 die Zeitauslösung dynamisch verändern. Wie weiter unten im Detail erläutert wird, erlaubt der Kostenmetrik-Betrieb des vorliegenden Computer-Drucker-Systems 200 dem Hostcomputer 202 die genaue Bestimmung der Zeitdauer, die der Drucker 218 benötigt, um eine Datenseite zu verarbeiten. Wie oben bemerkt, erlaubt der auf den Ressourcen basierende Ansatz für das Drucken dem Drucker 218 des Computer-Drucker-Systems 200 das Verarbeiten einer Seite in nicht mehr als dreißig Sekunden. Die meisten Seiten brauchen jedoch weniger als dreißig Sekunden. Der Hostcomputer 202 kann die Zeitauslösung auf die tatsächlich erwartete Verarbeitungszeit statt auf die statische Zeitauslösung von dreißig Sekunden setzen. Dabei ist zu beachten, daß diese Zeitdauer nicht die Zeit umfaßt, die der Druckermechanismus 226 (siehe Fig. 2) für das Drucken der Seite benötigt. Die Verarbeitungszeit umfaßt die Zeit, die der Drucker 218 benötigt, um die RPLs zu einer Bitmap für eine bestimmte Seite umzuwandeln.

Der Hostcomputer 202 kann zum Beispiel bestimmen, daß der Drucker 218 genau 11,5 Sekunden braucht, um eine bestimmte Seite zu verarbeiten. In Übereinstimmung mit den Prinzipien der vorliegenden Erfindung verändert der Hostcomputer 202 dynamisch die Zeitsperre für diese Seite auf etwas mehr als 11,5 Sekunden. Wenn der Drucker 218 die Kommunikation mit dem Hostcomputer nach zum Beispiel 11,6 Sekunden nicht wieder aufnimmt, weiß der Hostcomputer, daß ein Fehler aufgetreten ist. Dementsprechend erzeugt der Hostcomputer 202 eine Fehlermeldung. Da die Zeitauslösung tatsächlich mit der Verarbeitungszeit in Beziehung steht, erzeugt der Hostcomputer 202 niemals eine falsche Fehlermeldung, wie das im Stand der Technik der Fall ist. Die Kostenmetrik erlaubt es dem Hostcomputer 202, die Zeitauslösung dynamisch für jede Seite zu verändern. Wie oben genannt, ist die Zeitauslösung niemals höher als dreißig Sekunden.

Wenn der Ressourcenassembler 208 wie oben beschrieben Ressourcen zusammenstellt, berechnet er die präzisen Kosten für jedes Zeichenelement in einer RPL. Durch das Bestimmen der Kosten für jedes Zeichenelement weiß der Ressourcenassembler 208 genau, wie lange der Drucker 218 braucht, um die RPL zu verarbeiten. Der Hostcomputer 202 paßt die Zeitauslösung dementsprechend dynamisch an. Das exakte Verfahren, mit dem der Ressourcenassembler 208 die Kosten jedes Zeichenelements bestimmt, ist weiter unten im Detail beschrieben.

Der Ressourcenassembler 208 und der Ressourcenplaner 216 kommunizieren bei normalem Betrieb miteinander. Wenn der Ressourcenausführer 224 eine RPL zu einer Bitmapdatei umwandelt, sollte der Drucker 218 nur für die durch den Ressourcenassembler 208 berechnete Zeitdauer nicht mit dem Hostcomputer 202 kommunizieren. Wenn der Drucker eine längere Zeitdauer nicht mit dem Hostcomputer kommuniziert, weiß der Hostcomputer, daß ein Fehler aufgetreten ist. Die Zeitauslösung wird also genau durch den Hostcomputer 202 bestimmt und dynamisch für jede RPL oder Seite angepaßt. Des Hostcomputer erzeugt deshalb niemals eine falsche Fehlermeldung auf der Basis der Zeitauslösung, da der Hostcomputer 202 genau weiß, wie lange die Zeitauslösung sein sollte.

Wenn der Drucker 218 mit mehreren Hostcomputern 202 in einem Netz verbunden ist, kontrolliert der Hostcomputer nicht direkt, wann Daten zu dem Drucker 218 gesendet werden. Ein Netzserver (nicht gezeigt) nimmt Daten aus dem Hostcomputer an und gibt die Daten an den Drucker 218 weiter. Deshalb muß der Netzserver und nicht der Hostcomputer 202 die Zeitauslösung kontrollieren. Wenn der Netzserver die Fähigkeit aufweist, die Zeitauslösung dynamisch zu verändern, kann der Hostcomputer 202 die berechneten Kosten für das Verarbeiten jeder Datenseite enthalten, wobei der Netzserver die Zeitauslösung in der oben besprochenen Weise für jede Seite anpaßt.

Wenn der Netzserver die Zeitauslösung nicht dynamisch verändern kann, könnend die statischen dreißig Sekunden verwendet werden, da keine Datenseite

mehr als dreißig Sekunden für die Verarbeitung benötigt. Die Zeitauslösung von dreißig Sekunden der vorliegenden Erfindung liegt unter der typischen Zeitauslösung und steht tatsächlich mit der im Drucker 218 vorgenommenen Verarbeitung in Beziehung. Wenn die Zeitauslösung von dreißig Sekunden überschritten wird, informiert der Netzserver den Hostcomputer 202, wobei der Hostcomputer eine korrekte Fehlermeldung erzeugt. Das bedeutet, daß der Hostcomputer 202 niemals eine falsche Fehlermeldung erzeugt.

Die vorstehende Beschreibung betrifft die Datenverarbeitung für einen Drucker, wobei die Prinzipien der vorliegenden Erfindung jedoch ebensogut für ein anderes Peripheriegerät anwendbar sind. Ein Modem zum Beispiel benötigt eine Zeitdauer, um eine Datei mit einer ausgewählten Baudrate zu verarbeiten. Der Ressourcenassembler 208 kann die Kostenmetrik verwenden, um die Zeitdauer zu berechnen, die das Modem benötigt, um die im Hostcomputer 202 enthaltene Datei zu verarbeiten. Der Ressourcenassembler 208 kann die Zeitauslösung dynamisch in Abhängigkeit von den berechneten Kosten für die Datei im Peripheriegerät verändern.

Die Kostenmetrik ist bei Banddruckern besonders nützlich, wo die Echtzeit-Beschränkungen größer sind. Drucker, die über ausreichenden Speicher verfügen, um eine ganze Seite von Bitmapdaten zu speichern, weisen keine Echtzeit-Beschränkungen auf. Die vorliegende Erfindung ist jedoch auch in einem Bandmodus nützlich, weil der Hostcomputer 202 weiterhin Teile der RPLs verarbeiten kann, die die Seite beschreiben. Wenn ein Drucker mit einem Computernetz verbunden ist, ist die Kostenmetrik nützlich, um die Abschlußzeit für den ganzen Druckauftrag vorauszusagen. Die folgende Beschreibung nimmt auf Details des Betriebs einer Drucktechnologie Bezug, wobei das erfinderische System und das Verfahren der Kostenmetrik jedoch auch in anderen Bereichen anwendbar ist, wo eine Kostenplanung nützlich ist, zum Beispiel bei der Planung eines Echtzeitverfahrens. Einige in Echtzeit operierende Systeme erfordern Planungsdaten, darunter die Kosten von Echtzeit-Aktivitäten. Die Kostenmetrik ist auch bei der Profilierung eines Codemoduls nützlich, um ein Kostenmodell aller

möglichen Eingabeparameter zu konstruieren. Auch Kompilierer können eine Kostenmetrik verwenden, um ihre Optimierungsstrategien zu betreiben. Die Drucktechnologie muß sich Echtezeit-Beschränkungen unterwerfen, die Kostenmetrik ist jedoch auch in Bereichen ohne Echtzeit-Beschränkungen anwendbar. Zum Beispiel bei einem Programm, dessen Ausführungszeit von den Eingabedaten abhängig ist, können die Systeme des Standes der Technik nur Eingabedaten bereitstellen und die Kosten für diese besonderen Eingabedaten bestimmen. Im Gegensatz dazu kann eine Kostenmetrik verwendet werden, um das Programm für alle möglichen Typen von Eingabedaten zu modellieren.

In der Drucktechnologie liegt der Schlüssel für die Kostenmetrik darin, daß man dem Hostcomputer 202 erlaubt, die Daten für die Beschreibung der gesamten Seite in kleinere Teile aufzuteilen und diese Daten so schnell wie möglich, aber innerhalb der durch den Drucker 218 zu handhabenden Datenrate zum Drucker 218 zu senden, oder aber den Hostcomputer 202 dazu zu verwenden, einen Teil in Bitmapdaten umzuwandeln, bevor er zum Drucker gesendet wird. Mit anderen Worten, kann der Hostcomputer 202 Teile der Daten für die Beschreibung der Seite verarbeiten und die verarbeiteten Daten zum Drucker 218 senden. Auch wenn die Teile für die Verarbeitung zum Drucker gesendet werden, ist die vorliegende Erfindung schneller als die Systeme des Standes der Technik, da die Daten in einer Form vorliegen, die die Verarbeitung vereinfacht.

Die Kostenmetrik sieht die Kosten für jedes Zeichenelement für eine bestimmte RPL oder ein bestimmtes Band vor. Diese Kosten werden durch den Ressourcenassembler 208 verwendet, um zu entscheiden, ob eine RPL direkt zum Drucker gesendet wird, die RPL zum Drucker 218 gesendet wird, um sie vorzuverarbeiten, bevor eine Echtzeit-Aktivierung vorgenommen wird, oder ob eine Bitmap der RPL im Hostcomputer 202 erzeugt wird und die Bitmap zum Drucker 218 gesendet wird. Diese drei Optionen erlauben das Verarbeiten von Daten in Zeiten, in denen der Drucker 218 nicht mit dem Drucken einer Seite beschäftigt ist (z.B. während des Einziehens von Papier). Das Ergebnis ist eine beträchtlich reduzierte Gesamtdruckzeit.

Das Aufteilen der Seite in kleinere Teile kann eine Bandverarbeitung umfassen. Wie zuvor beschrieben, umfaßt die Bandverarbeitung das Aufteilen der gedruckten Seite zu einer Anzahl von horizontalen Segmenten, die als Bänder bezeichnet werden. Ohne Bandverarbeitung benötigen Drucker große Speichermengen, um eine ganze Seite in der Form einer Bitmap zu speichern. Mit der Bandverarbeitung braucht der Drucker nur die Bitmap für ein einziges Band zu speichern, wobei der Drucker jedoch auch mehr als ein Band speichern kann, wenn er über ausreichenden Speicher verfügt.

Fig. 4 stellt einige der möglichen Optionen für das Senden eines Datenbandes zum Drucker dar. Die A-Form in Fig. 4 stellt ein Band dar, das unter Verwendung einer Anzahl von Zeichenelementen in der Form einer RPL beschrieben wird. Die B-Form der Daten gibt eine Bitmapdatei wieder, die das Band beschreibt. Dabei ist zu beachten, daß die Bitmap entweder im Hostcomputer 202 oder im Drucker 218 erzeugt werden kann. In einigen Fällen, kann der Hostcomputer 202 die Bitmapdatei komprimieren, wie in der C-Form dargestellt. Wie aus Fig. 4 entnommen werden kann, erlaubt es die Option 1 der Datenverarbeitung dem Hostcomputer 202, die RPL (A-Form) in eine Bitmapdatei (B-Form) umzuwandeln und die Datei dann zu komprimieren (C-Form). Der Ressourcenassembler 208 dient dabei als Host-Ressourcenausführer (HRE), um die RPL zu einer Bitmap umzuwandeln. Die komprimierte Datei (C-Form) wird zum Drucker 218 übertragen und im Drucker 218 zurück in eine Bitmap (B-Form) dekomprimiert. Alternativ dazu, überträgt der Hostcomputer 202 die RPL (A-Form) direkt zum Drucker 218. Der Drucker verarbeitet die RPL (A-Form), um eine Bitmapdatei (B-Form) im Drucker zu erzeugen. Als eine dritte Alternative verarbeitet der Hostcomputer 202 die RPL (A-Form) zu einer Bitmapdatei (B-Form) und überträgt die Bitmapdatei zum Drucker, ohne die Daten zu komprimieren. Der Drucker 218 empfängt die Bitmapdatei (B-Form), die keine zusätzliche Verarbeitung mehr erfordert, außer natürlich für das Drucken.

Die Optionen 1 und 3 umfassen die Verarbeitung durch den Hostcomputer 202. Die für die Verarbeitung eines Bandes verfügbare Zeit hängt von der Druckergeschwindigkeit und dem Druckerspeicher 222 ab. Jedes Band hat nur eine begrenzte Zeitdauer, in der sie durch den Drucker 218 verarbeitet werden muß. Wenn der Ressourcenassembler 208 bestimmt, daß das Band nicht in der verfügbaren Zeitdauer verarbeitet werden kann, wandelt der Hostcomputer 202 das Band in eine Bitmapdatei um. Alternativ dazu kann der Hostcomputer 202 die RPL zum Drucker 218 senden, um die RPL vorzuverarbeiten, bevor eine Echtzeit-Aktivierung vorgenommen wird. Die Fähigkeiten für eine Vorverarbeitung hängen von der Berechnungsleistung des Druckers 218 und von der Speichermenge des Druckerspeichers 222 ab, die für das Speichern der Bänder von Bitmapdaten verfügbar ist. Bänder, die zu zeitaufwendig sind, um durch den Drucker 218 verarbeitet zu werden, werden als komplexe Bänder bezeichnet. Komplexe Bänder überschreiten die erlaubte Druckzeit, was bedeutet, daß das Band nicht schnell genug gedruckt werden kann, um die Echtzeit-Beschränkungen zu erfüllen. Wenn ein Band komplex ist, verarbeitet der Hostcomputer 202 das Band unter Verwendung der in Fig. 4 gezeigten Option 1 oder 3.

In der Praxis wird das Verfahren durch die Tatsache verkompliziert, daß eine zusätzliche Zeit für das Verarbeiten eines Bandes verfügbar sein kann, während sich das Papier durch den Drucker 218 bewegt. Es kann auftreten, daß ein Band effektiv in Echtzeit verarbeitet wird, wenn die Zeit, während der sich das Papier bewegt, berücksichtigt wird. Die vorliegende Erfindung schließt diesen Faktor bei der Bestimmung ein, ob der Drucker die Bänder in Echtzeit verarbeiten kann oder nicht. Nur wenn „zu viele" Bänder auf einer Seite nicht in Echtzeit verarbeitet werden können, verarbeitet der Hostcomputer 202 eines der Bänder. Viele Faktoren werden bei der Bestimmung einbezogen, wie viele Bänder „zu viel" sind: zum Beispiel die Berechnungsleistung des Druckers 218, die Geschwindigkeit des Druckermechanismus 226 (siehe Fig. 2) und die verfügbare Speichermenge des Druckerspeichers 222. In der vorliegenden bevorzugten Ausführungsform weist jedes Band auf einer Seite 208 Scanzeilen auf. Die Anzahl der Bänder auf einer Seite hängt also von der Seitenlänge ab. Dabei sollte jedoch beachtet werden, daß die in der vorliegenden

Ausführungsform ausgewählte Bandgröße kein einschränkender Faktor ist, sondern willkürlich ausgewählt werden oder sogar von Band zu Band variieren kann. Die wie auch immer bestimmte Bandgröße muß jedoch im voraus bekannt sein, um zu bestimmen, ob ein Band einfach oder komplex ist.

Der Hostcomputer 202 analysiert die Komplexität der Bänder, indem er die für die Ausführung der Optionen 1 oder 3 erforderliche Zeit plus der für die Übertragung der Daten zum Drucker 218 erforderliche Zeit bestimmt und die Gesamtzeit mit der für die Ausführung der Option 2 plus der für die Übertragung der Daten zum Drucker 218 erforderlichen Zeit vergleicht. Der Hostcomputer 202 wählt die Option mit der kürzesten Gesamtzeit.

Es besteht eine Obergrenze für die Anzahl der komplexen Bänder, die auf einer Seite auftreten können, bevor der Hostcomputer 202 einfach die gesamte Seite verarbeitet. Offensichtlich variiert dieses Limit in Abhängigkeit von den relativen Berechnungsleistungen des Hostcomputers 202 und des Druckers 218 sowie von der verfügbaren Speichermenge des Druckerspeichers 222. Wenn dieses Limit nicht erreicht ist, verarbeitet der Hostcomputer 202 nur die Bänder, die durch den Drucker 218 nicht in Echtzeit verarbeitet werden können. Es wird eine Kostenmetrik verwendet, um zu bestimmen, ob ein Band als einfach oder komplex ist.

Wie am besten in Fig. 5 gesehen werden kann, erstellt das Kostenmetriksystem Ausführungszeiten für eine Vielzahl von Zeichenelementen unter Verwendung verschiedener Rasteroperationscodes (ROPs). Die Zeichenelemente werden als ein Testbefehlssatz 250 gespeichert. Der Testbefehlssatz kann alle möglichen Zeichenelemente umfassen oder eine Teilmenge aller möglichen Zeichenelemente. Die ROPs sind ein Satz von Funktionen, die die Art und Weise definieren, in der Bits in einer Quellcode-Bitmap mit den Bits in einer Brush-Bitamap und den Bits in einer Ziel-Bitmap kombiniert werden.

In der vorliegenden Ausführungsform der Erfindung sind 256 verschiedene ROPs vorgesehen, die durch acht Datenbits wiedergegeben werden. Die ROPs

können je nach der Anzahl der empfangenen Argumente in drei Typen gruppiert werden. Diese Typen von ROPs sind in der folgenden Tabelle 1 gezeigt.

ROP Typen	Verwendete Argumente
2	Ziel, Muster
3	Ziel, Muster, Quellen-Bitmap
4	Ziel, Muster, Quellen-Bitmap, Maske

Tabelle 1. ROP-Typen

Das Computer-Drucker-System 200 berechnet eine Reihe von Ausführungszeiten für verschiedene Zeichenelemente, wenn der Drucker 218 auf dem Computer 202 installiert ist. Alternativ dazu kann die Kostenmetrik vorbestimmt sein, wobei die Kosteninformation für die besonderen Drucker als eine Echtzeit-Binär-Datei für den Ressourcenassembler 208 bereitgestellt wird. Das erfinderische System kann auch verwendet werden, um die Leistung eines Druckers zu bewerten. Ein Timer 252 wird verwendet, um die Ausführungszeiten für die verschiedenen Zeichenelemente zu messen. Durch das Messen der Ausführungszeiten für den Testbefehlssatz 250, erstellt ein Bewertungsmodul 254 ein Druckermodell 256, das den besonderen Drucker 218 wiedergibt. Das Druckermodell 256 wird verwendet, um eine Protokolldatei 260 zu erzeugen, die die Kostenmetrikdaten für die besondere Kombination aus dem Hostcomputer 202 und dem Drucker 218 enthält. Um die Kostenmetrikdaten zu vereinfachen, werden die Daten gruppiert und in einem Satz von Kostentabellen 262 gespeichert. Die Kostentabellen 262 enthalten die Kostenmetrikdaten für bestimmte Klassen von Zeichenelementen. Dies erfordert wesentlich weniger Speicher als eine Datei, die komplette Kostenmetrikdaten für alle möglichen Zeichenelemente enthält. Ein Satz von Map-Tabellen 264 sieht eine einfache Einrichtung zum Zugreifen auf die Kostendaten in den Kostentabellen 262 vor. Die Kostentabellen 262 und die Map-Tabellen 264 sind mit dem Ressourcenassembler 208 verbunden, der die Kostendaten verwendet, um die Kosten jeder RPL zu dem Zeitpunkt, zu dem ein Dokument gedruckt werden soll, zu bestimmen. Auf diese Weise können die vorbestimmten Kostenmetrikdaten

verwendet werden, um die Datenverarbeitungsverantwortlichkeit zwischen dem Hostcomputer 202 und dem Drucker 218 dynamisch zu teilen.

Der Timer 252 wird verwendet, um die Kosten der Ausführung des Testbefehlssatzes 250 zu bestimmen. In der vorliegenden Ausführungsform verwendet der Timer 252 die interne Zeitrechnungsfähigkeiten, die bei typischen Computern verfügbar sind. Wie in Fig. 6 gezeigt, verwenden typische Computer einen Oszillator 400 mit einer Frequenz von ungefähr 1,193,180 Hertz. Die Periode beträgt bei dieser Frequenz 838,0965 Nanosekunden. Für die vorliegende Beschreibung soll der Taktzyklus mit 838,0965 Nanosekunden definiert werden. Ein programmierbarer Intervallzähler (PIC) 402, der einen 16-Bit-Zähler enthält, dekrementiert bei jedem Taktzyklus um zwei. Der PIC 402 wird durch das Computerbetriebssystem programmiert, um kontinuierlich rückwärtsschreitend von 655355 zu 0 zu zählen, wobei der PIC 402 dann die Interrupt-Leitung 404 zur CPU 406 schaltet. Die CPU 406 antwortet nur auf die Anstiegsflanke der Interrupt-Leitung. Da jedoch der PIC 402 bei jedem Taktzyklus um zwei dekrementiert, wird ungefähr alle 55 Millisekunden ein Interrupt durch den PIC 402 erzeugt. In Antwort auf den Interrupt inkrementiert die CPU 406 den Wert des Taktspeicherbereichs 408. Durch die Verwendung der kombinierten Zähler im PIC 402 und des Taktspeicherbereichs 408 ist es möglich, einen 48-Bit-Zähler mit einer Auflösung von 838,0965 Nanosekunden zu erhalten.

Die vorliegende Erfindung führt wiederholt dasselbe Zeichenelement für eine vorbestimmte Zeit 410 aus und zählt, wie oft das Zeichenelement ausgeführt wurde. Ein Teiler 412 teilt die abgelaufene Zeit, die durch die interne Zeitberechnungsschaltung des Computers gemessen wurde, durch die Anzahl der ausgeführten Zeichenelemente, um die Kosten pro Zeichenelement zu berechnen.

Die Zeit auf dem Hostcomputer 202 (siehe Fig. 2) kann am einfachsten gemessen werden, indem die Datum/Zeit-Funktion in DOS aufgerufen wird. Dies ist der allgemein verwendete Ansatz. Ein anderes Verfahren besteht darin, eine BIOS-Funktion aufzurufen, um die Anzahl der Taktzyklen à 55 Millisekunden festzustellen, die seit Mitternacht abgelaufen sind. Dieses Verfahren ist jedoch problematisch,

wenn das Programm nachts läuft, da das Mitternachtsereignis übersehen werden kann und das Datum/Zeit einen Tag zu spät sein kann. Die vorliegende Erfindung verwendet einen dritten Ansatz. Da das erfinderische System eine feinere Auflösung als 55 Millisekunden erfordert, verwendet es den Hardwaretimer 252 (siehe Fig. 5) auf dem Hostcomputer 202, um die Zeitdauer zu decodieren, die für die Verarbeitung der Zeichenelemente verwendet wird. Insbesondere wird der programmierbare Intervall-Controller-Check (PIC) 402 (siehe Fig. 6), der die zuvor genannte BIOS-Funktion betreibt, für die Zeitmessung verwendet. Das erlaubt es dem System, einen Timer mit einer viel feineren Auflösung zu erhalten. Anstatt von 55 Millisekunden ist es möglich, eine theoretische Auflösung von ungefähr 838 Nanosekunden zu erhalten, da der Hardwaretimer ein 16-Bit-Timer ist und (55 Millisekunden)/65536 gleich 838 Nanosekunden ist. Es ist jedoch zu beachten, daß die Kommunikationsverbindung mit dem Drucker eine gewisse Willkür aufweist und in der Praxis nur eine Auflösung von 10 Mikrosekunden erhalten werden kann. Alle oben angegebenen Zahlen sind gerundete Zahlen. Der Timer 252 ist viel präziser als diese gerundeten Zahlen angeben. Die vorliegende Erfindung verwendet eine Kombination aus dem BIOS-Zähler und dem PIC, um den für die Messung der Ausführzeiten der Zeichenelemente verwendeten Timer 252 zu erhalten.

Der PIC 402 weist drei Countdown-Timer (nicht gezeigt) auf, die mit einer konstanten Rat getaktet sind. Die PIC-Chip-Zähler verwenden einen Wert von 65536 und zählen rückwärtsschreitend. Wenn die Zähler auf dem PIC null erreichen, starten sie einen Prozeß. Der Timer 0 erzeugt einen Interrupt (Vektor 8H, 20-23H). Der Originalwert wird in den Zähler geladen und der Prozeß fährt fort. BIOS inkrementiert eine ULONG-Zählung an der Lokation 40H:6CH. Der Counter gibt die Anzahl der Ticks wieder, die seit dem letzten Mitternachtsereignis (12-Uhr-Mitternacht) aufgetreten sind (oder aufgetreten sein sollten). Der BIOS-Zähler wird durch den Timer-Tick, die Echtzeit-Uhr (beim Booten) und durch den MS-DOS-Befehl „TIME", wenn die Echtzeit-Uhr eingestellt wird, aktualisiert. Um Mitternacht wird der BIOS-Zähler rückgesetzt und startet wieder von null. 1537040 Timer-Ticks entsprechen einem Zeitraum von 24 Stunden. Der BIOS.-Zähler kann Werte von 0 bis 1573039 enthalten.

Es gibt einen BIOS-Aufruf, um diesen Zähler zu erhalten, der aber in dieser Anwendung nicht verwendet werden kann, weil diese Bedingung bei Auftreten eines „Mitternachtsereignisses" zum Aufrufer zurückgegeben und vergessen wird. Die Handhabung des „Mitternachtsereignisses" ist ein schwieriger Prozeß, der in der vorliegenden Erfindung vermieden wird. Normalerweise werden Überlaufbedingungen durch MS-DOS gehandhabt. Wenn der BIOS-Aufruf verwendet wird und die Überlaufbedingung auftritt, gibt es keine Möglichkeit, um dieses Ereignis zu MS-DOS zurückzuführen, außer das Datum um einen Tag vorzustellen. Die dafür erforderliche Logik ist zu komplex.

Statt dessen betrachtet der Timer 252 einfach den BIOS-Bereich und liest den 4-Byte-Tick-Zähler, wobei jedoch zu beachten ist, daß der Tick-Zähler um Mitternacht auf Null zurückgesetzt wird. Es ist wichtig, MS-DOS zu einer bestimmten Aktion aufzurufen, so daß es diesen BIOS-Aufruf wenigstens alle 24 Stunden vornehmen muß. Zum Beispiel kann eine Datei geschrieben werden, da MS-DOS dabei das Änderungsdatum der Datei aktualisieren muß.

Das Synchronisieren des Hostcomputers 202 mit dem Drucker 252 für die Messung der Kosten, schließt die Verwendung des Timers 0 (nicht gezeigt) des PIC 402 ein. Der Timer 0 betreibt die Systemuhr. Der Timer 0 ist programmiert, um in Modus 3 zu operieren - Rechteckwellengenerator. Das bedeutet, daß die Ausgabe hoch bleiben wird (erste Phase) bis die Hälfte der Zählung abgeschlossen ist (für die geraden Zahlen) und für die andere Hälfte der Zählung niedrig geht (zweite Phase).

Wenn die Zahl ungerade ist und die Ausgabe hoch, dekrementiert der erste Taktimpuls (nach dem Laden der Zählung) die Zählung um 1. Darauffolgende Taktimpulse dekrementieren den Takt um 2. Nach einem Zeitablauf geht die Ausgabe niedrig und die volle Zählung wird neu geladen. Der erste (auf das neue Laden folgende) Taktimpuls dekrementiert den Zähler um 3. Die darauffolgenden Taktimpulse dekrementieren die Zählung um 2 bis zum Zeitablauf. Dann wird der ganze Prozeß wiederholt. Wenn auf diese Weise die Zählung ungerade ist, bleibt die

Ausgabe für (N+1)/2 Zählungen hoch und für (N-1)/2 Zählungen niedrig. Die Zählung kann unter Verwendung von PIC_LATCHIT gelesen werden, wobei die Zählung zwischengespeichert und dann gelesen wird. Das einfache Lesen des Wertes gibt jedoch nicht an, ob der Timer 0 in der ersten Phase oder in der zweiten Phase des Zyklus ist. Die einzige Möglichkeit, um dies zu erfahren, besteht darin zu warten, bis der Zähler übergeht, und zu prüfen, ob ein Takt-Tick erzeugt wird.

Der Rechteckwellengenerator (Timer 0) ist mit dem Interrupt-Vektor 8H (Vektoradresse 20-23H) assoziiert. Dieser zeigt normalerweise in das BIOS, das auch einen anderen Timer-Tick-Interrupt-Vektor bei ICH (Vektoradresse 70-73H) aufruft. Normalerweise zeigt ICH auf ein DUMMY_RETURN. Wenn der Vektor 8H einfach ersetzt wird, funktioniert die Systemuhr nicht richtig. Der Vektor 8H muß alle 18,2 Millisekunden aufgerufen werden.

Es sind zwei 16-Bit-Werte im BIOS-Bereich und ein 16-Bit-Wert im PIC-Chip für insgesamt drei USHORTs vorgesehen. Jeder USHORT-Abruf ist elementar, wobei aber ein Interrupt zwischen jedem dieser Abrufe auftreten kann, so daß keine Möglichkeit besteht, alleine unter Verwendung dieser Punkte einen sauberen 48-Bit-Elementarzugriff zu erzeugen. Wenn zum Beispiel ein Interrupt zwischen dem ersten Abruf und dem zweiten Abruf auftritt, kann man 000BFFFF an Stelle des korrekteren 000AFFFF oder 000B0000 lesen. Die beiden letzten sind annehmbar, das erste ist jedoch zu weit weg und würden natürlich zu einem fehlerhaften Zeitablauf führen.

Da der Timer-Interrupt nur alle 18,2 Millisekunden auftritt, kann man den BIOS-Bereich zweimal lesen und einen Wert ausgeben, der wiederholbar ist (man kann auch die Interrupts deaktivieren oder einen 23-Bit-Zugriff unter Verwendung von 386 oder 486 Registern vornehmen, wobei aber die Deaktivierung nicht den PIC-Zähler stoppt).

Um dieses Problem zu lösen, wird zuerst ein sicherer 32-Bit-Zahlenzugriff für den BIOS-Bereich vorgenommen. Da der BIOS-Bereich nur alle 18,2 ms aktualisiert wird, kann man ihn einfach zweimal ablesen und sicherstellen, daß er sich nicht

verändert hat. Die aus dem BIOS ausgelesenen Werte sind ulBIOS1 und ulBIOS2. Die aus dem PIC-Zähler ausgelesenen Werte sind usPic1 und usPic2, usw.

Die Prozedur für das Zugreifen auf den PIC 402 besteht darin, zuerst die Absichten in PIC_CONTROL_PORT zu schreiben und dann ein oder zwei Bytes des entsprechenden TIMERx-Ports (auf dem PIC) zu lesen oder in diesen zu schreiben.

In der vorliegenden Anwendung ist es wichtig, daß die Differenz zwischen dem Start und dem Stop genau gemessen wird. Ob einige wenige Millisekunden vor oder nach dem Test verschwendet werden, ist nicht sehr wichtig. Um dies zu bewerkstelligen, wartet man, bis der Timer „übergeht", bevor zur Anwendung zurückgekehrt wird. Wenn dann der Stop-Timer-Aufruf gemacht wird, verfügt man über eine genaue Messung der Zwischenzeit. Da der PIC um 2 dekrementiert wird, ist das niedrigstwertige Bit immer null und deshalb bedeutungslos. Wenn der PIC 402 null erreicht, wird OUT0 geschaltet und der PIC neu geladen. Interrupts treten nur bei jedem zweiten OUT0-Übergang auf. Es sind also nur die 15 höchsten Bits des PIC bedeutend, wobei ein anderes Bit (OUT0) nicht verfügbar ist und durch Detektivarbeit abgeleitet werden muß.

Die einzige Möglichkeit zum Beobachten dieses versteckten Bits besteht darin, zu warten, bis der PIC-Zähler übergeht, und dann zu sehen, ob sich die 32-Bit-Zählung im Zähler verändert. Wenn also das versteckte Bit '1' ist, wird der Übertrag weitergeleitet, andernfalls wird der Übertrag nicht weitergeleitet. Es muß eine gewisse zusätzliche Vorsicht walten gelassen werden, da der 15-Bit-PIC-Zähler ein Hardware-Zähler ist. Er kann nicht gestoppt werden und läuft asynchron zum Programm. Durch den Hardware-Zähler veranlaßte Interrupts werden nicht unmittelbar durch die CPU verarbeitet. Es ist also eine Verzögerung gegeben, wobei diese Verzögerung nicht voraussagbar ist.

Der Timer 252 kann auch ohne die anderen Aspekte der Kostenmetrik zum Messen der Codeausführungszeiten verwendet werden. Das ist besonders bei

Anwendungen wie dem Profilieren der Codes zum Messen der Ausführungszeiten nützlich.

Das Bewertungsmodul 254 verwendet die Timerinformation, um die Kostenmetrik zu erhalten. Die Zielsetzung des Bewertungsmoduls 254 ist es, die Kosten einer einzigen Zeichenprimitive zu bestimmen. Wie zuvor erläutert, ist die Auflösung typischer Timer für die Zwecke der vorliegenden Erfindung nicht ausreichend. Die Timing-Auflösung über den Kommunikationskanal zum Drucker weist eine wiederholbare Auflösung von ungefähr 10 Mikrosekunden auf, wobei aber die erforderliche Auflösung bei weniger als 1 Nanosekunde liegt. Um die erforderliche Auflösung zu erhalten, wird der Drucker 218 aufgefordert, eine Anzahl von Zeichenelementen auszuführen. Die Anzahl der Elemente kann 10,000 überschreiten.

Das Bewertungsmodul 254 beginnt, indem es den Drucker 218 auffordert, einige Zeichenelemente auszuführen. Dabei ist zu beachten, daß dasselbe Zeichenelement mit denselben Parametern, demselben ROP, derselben Form und demselben Brush verwendet wird. Die Anzahl der ausgeführten Elemente wird erhöht, bis eine vorbestimmte Zeitspanne 410 abgelaufen ist. Das Bewertungsmodul 254 bewertet die Anzahl der während der vorbestimmten Zeitdauer ausgeführten Zeichenelemente, um die Zeitdauer zu berechnen, die für das Ausführen des besonderen Elements unter Verwendung des besonderen ROP, der besonderen Form und dem besonderen Brush erforderlich ist. Mit anderen Worten bestimmt das Bewertungsmodul 254, wie viele Zeichenelemente in der vorbestimmten Zeitdauer ausgeführt werden, um zu bestimmen wie lange es dauert, um ein einziges Zeichenelement dieses Typs auszuführen. Diese Prozedur für ein besonderes Zeichenelement wird als Sample bezeichnet.

Das Bewertungsmodul 254 bestimmt Samples für verschiedene Typen von Zeichenelementen. In der vorliegenden bevorzugten Ausführungsform wählt das Bewertungsmodul 254 eine vorbestimmte Zeitdauer von ungefähr 300 Millisekunden. Mit einer Timer-Genauigkeit von 10 Mikrosekunden resultiert daraus ein maximaler Fehler von ungefähr 0,0033%. Die für das Bestimmen eines Samples erforderliche

Zeit ist relativ konstant. Wenn die Zeichenelemente einfach sind, muß das Bewertungsmodul 154 in der vorbestimmten Zeitdauer einige Zeichenelemente mehr senden. Wenn die Zeichenelemente relativ komplex sind, sendet das Bewertungsmodul während der vorbestimmten Zeitdauer weniger Zeichenelemente. Die einzige Variable ist also die Anzahl der Zeichenelemente, die während der vorbestimmten Zeitdauer gesendet wird. Wie leicht erkannt werden kann, kann praktisch jede Auflösung erreicht werden, indem die vorbestimmte Zeitdauer 410 variiert wird. Die Verwendung von zum Beispiel einer Testzeitdauer von ungefähr 3000 Millisekunden (3 Sekunden) ergibt eine Auflösung von 0,00033%. In der vorliegenden bevorzugten Ausführungsform wurde eine vorbestimmte Zeitdauer von ungefähr 300 Millisekunden gewählt, um die für die Ausführung des Testbefehelssatzes 250 erforderliche Zeit zu minimieren.

Das Druckermodul 256 weist das Bewertungsmodul 254 an, welche Zeichenelemente aus dem Testbefehlssatz 250 zum Drucker 218 zu senden sind. Das Druckermodul 256 verwendet die Sampledaten, um ein Modell für den besonderen beim Hostcomputer 202 installierten Drucker 218 zu erstellen. Das Modell für einen besonderen Drucker wird entwickelt, indem die Kosten für das Ausführen verschiedener Funktionen bestimmt werden. Es können 1, 5, 10 oder mehr Samples erforderlich sein, um die Kosten der verschiedenen Funktionen zu bestimmen. Das Druckermodell 256 kann eine Statistik für die am wenigsten linearen Quadratregression (LLSR = Least-Linear-Square Regression) verwenden, um den Anstieg oder die Koordinatenstrecke bestimmter Kostenkurven zu finden. Diese Techniken sind dem Fachmann wohlbekannt und werden hier nicht ausführlicher beschrieben. Einfach ausgedrückt, ist die LLSR ein statistisches Verfahren, um eine Linie auszuwählen, die einer Reihe von Punkten „am besten" angepaßt ist. Die Linie ist ein Modell der Punkte. Wenn das Modell „gut" ist, sind die Punkte sehr nahe an der Linie. Die vorliegende Erfindung verwendet die LLSR in verschiedener Weise. Zum Beispiel verwendet das Bewertungsmodul 254 die LLSR, um die Kosten pro Zeichenelement herauszufinden. Das Druckermodell verwendet die LLSR, um sich den Daten für die Zeichenelemente anzupassen, indem alle bis auf eine variable Konstante gehalten werden. Dies erlaubt es dem Druckermodell 256, den Effekt

dieser Variable von den anderen zu isolieren. Die Anwendung der LLSR für das Druckermodell 256 wird im folgenden ausführlicher erläutert.

Allgemein sind mit allem, was der Drucker tut, Kosten verbunden. So gibt es zum Beispiel die Kosten für das Verarbeiten eines Elements zu einer Bitmapdatei für ein bestimmtes Band. Außerdem gibt es die Kosten für das Markieren eines bestimmten Punktes auf der Seite, und die Kosten für das Ausfüllen einer Reihe auf der Seite. Die Kosten für das Verarbeiten eines Elements werden mit P angegeben, während die Kosten für einen Punkt und eine Reihe jeweils mit D und R angegeben werden. Das Druckermodell 256 bestimmt die Kosteninformation zum Ausführen verschiedener Zeichenelemente mit verschiedenen Dimensionen. Ein Element mit einer Null-Dimension weist nur einen Typ von zu kennzeichnenden Kosten auf. Andere Elemente weisen eine oder mehrere Dimensionen und einen oder mehrere zu kennzeichnende Typen von Kosten auf. Tabelle 2 zeigt die Zeichenelemente und die mit ihnen verbundenen Kosten.

Zeichenelement	**Anzahl der Dimensionen im Element**	**Damit verbundene Kosten**
Linie	0	Element-Kosten
Glyph	1	Element- und Punktkosten
Rechteck, Keil	2	Element-, Punkt- und Reihenkosten
GlyphBDN	3	Element-, Punkt-, Reihen- und Mehrfachglyphen-Kosten

Tabelle 2. Zeichenelemente und damit verbundene Kosten

Elemente der Null-Dimension weisen dieselbe Ausführungszeit auf, unabhängig davon ob es sich um einen ROP-, Brush-, Zeichenpositions- oder Stiftstil handelt. Beispiele für Null-Dimension-Elemente sind Befehle für die Cursorbewegung wie SetRowAbsolute (SetRowAbs) oder SetColumnAbsolute (SetColAbs). Das SetRopAndBrush-Element ist auch ein Element der Null-Dimension, das aber

einen einmaligen Fall darstellt, weil es 1024 mal numeriert werden muß, da es vier mögliche Brushes und 256 mögliche ROPs gibt (4x256=1024).

Eindimensionale Elemente hängen von einer einzigen Variable, wie der Breite oder der Länge ab. Wie zuvor erwähnt, verwendet das Druckermodell 256 LLSR für die Bestimmung der am besten passende Linie, um die mit dem Element assoziierten Parameter zu kennzeichnen. Es sind allgemein fünf Samples erforderlich, um die am besten passende Linie zu bestimmen. Ein Beispiel für ein eindimensionales Element ist LineAbsolute (LineAbsS), das ein Element zum Zeichen einer Linie ist.

Zweidimensionale Elemente hängen von zwei Variablen, wie der Breite und der Höhe ab. Es sind allgemein zehn Samples erforderlich, um diese Elemente zu kennzeichnen. Ein Beispiel für ein zweidimensionales Zeichenelement ist Rectangle (Rect), das ein Rechteck zeichnet.

Es gibt nur ein dreidimensionales Element (GlyphBDN). Dieses Element erfordert allgemein fünfzehn Samples, um es adäquat zu kennzeichnen. Das GlyphBDN-Element ist von den drei Variablen Höhe, Breite und Anzahl der Unterglyphen abhängig. Ein Unterglyph ist ein Glyph, der in dem GlyphBDN-Zeichenelement enthalten ist. Während Glyph-BDN als ein einfaches Zeichenelement betrachtet wird, können mehrere Unterglyphen mit jeweils einer eigenen Breite und Höhe vorgesehen sein. Die Kosten für das Zeichnen eines GlyphBDN-Elements variieren je nachdem, ob es für das Zeichen eines Glyphen oder mehrerer Glyphen verwendet wird.

Die vorliegende bevorzugte Ausführungsform macht mehrere Annahmen über den Betrieb eines besonderen Druckers. Diese Annahmen reduzieren die Anzahl der für verschiedene Elemente erforderlichen Modelle. Insbesondere nimmt die Erfindung an, daß die Kosten für das Verarbeiten eines besonderen Elements (P), die Punktkosten (D) und die Reihenkosten (R) für einen gegebenen Drucker

konstant sind. Das erlaubt die Kennzeichnung der verschiedenen Dimensionen von Elementen durch die Gleichungen in Tabelle 2 unten.

Dimension	Ausführungsmodell
0	$t = P$
1	$t = P+wR$
2	$t = P+wR+whD$
3	$t = P+nN+\sum_{i=1}^{n} (w_iR+w_ih_iD)$

Tabelle 3. Tabelle des Ausführungsmodells für Elemente der verschiedenen Dimensionen

wobei T die erforderliche Gesamtzeitdauer ist, P die für die Ausführung des Elements erforderliche Zeitdauer ist, h die Höhe des Objekts ist, R die Reihenkosten sind, w die Breite des Objekts ist, D die für das Berühren eines Punkts erforderliche Zeit ist, n die Anzahl der Teilelemente eines Elements ist und N die Kosten für das Verarbeiten eines Teilelements ist.

Um die Kosten für ein Element zu bestimmen, wählt das System eines der Modelle aus, das der Dimension des zu verarbeitenden Elements entspricht. Um zum Beispiel die Kosten eines zweidimensionalen Elements zu bestimmen, verwendet das System zwei Phasen der Berechnung.

In der Phase eins soll der Parameter D isoliert werden. Um dies zu bewerkstelligen, ist es möglich, die LLSR zu verwenden und w konstant zu halten. Indem nur h variiert wird, kann der Parameter D isoliert werden.

Unter Verwendung der Formel für die zwei Dimensionen aus Tabelle 3, können die Gesamtkosten für ein Zeichenelement durch die folgende Formel erhalten werden.

$$t = P+wR+whD$$

Nehme an, daß

$$K_1 = P+wR$$
$$K_2 = wD$$

Dann

$$t = K_1+K_2h$$

Durch das Nehmen von fünf Samples mit verschiedenem h und unter Verwendung von LLSR kann das System die zwei Konstanten finden. Da der Wert von w bekannt ist, ist es möglich, D zu bestimmen.

In Phase zwei soll der Parameter R isoliert werden. Um dies zu bewerkstelligen, ist es erforderlich h konstant zu halten und w zu variieren. Die Gleichungen sind nun:

$$t = P+wR+whD$$

wobei w variiert wird. Um einen Satz von Samples zu bilden, der nur in w variiert, ist es erforderlich, den whD-Term von jedem Sample zu subtrahieren. Daraus resultiert:

$$t_{llsr} = t_{sample} - whD$$

wobei t_{llsr} die Zeitdauer ist, die durch die LLSR-Analyse verwendet wird und wobei t_{sample} die Samplezeit ist, die durch das Bewertungsmodul 254 bestimmt wird. Daraus resultiert eine einfache Gleichung, die leicht für die Daten in der LLSR-Analyse angewendet werden kann. Es ist möglich, dies zu machen, da D aus Phase 1 bekannt ist und auch w und h für jedes Sample bekannt sind.

Nehme an, daß

$K_1 = P$

$K_2 = R$

Dann

$t = K_1 + K_2 w$

Durch das Nehmen von fünf Samples aus verschiedenen w und unter Verwendung von LLSR können die beiden Konstanten gefunden werden. Die zwei Konstanten sind P und R. Die zwei Phasen erlauben also die Bestimmung der Parameter P, D und R. Die Kosten für das Element können dann für jede beliebige Breite und Höhe bestimmt werden.

Die Anzahl der Dimensionen beim Verarbeiten eines Elements bedingt die Anzahl der Tests in einem Sample und damit die pro Sample erforderliche Gesamttestzeit. Um die Ausführungskosten für alle Zeichenelemente zu berechnen, verwendet das System 1, 5, 10 oder 15 Testpunkte für die Höhe und die Breite während Phase eins und zwei. Die ausgewählten Werte haben höhere Ausführungszeiten von 300 bis 1500 Millisekunden zur Folge, je nach der Anzahl der ausgewerteten Testpunkte. Tabelle 4 zeigt die Beziehung zwischen den Dimensionen und der Testzeit. Um die Testzeit zu minimieren, kann für jeden Testpunkt eine Anzahl von kleinen Zeichenelementen durch das Bewertungsmodul zum Drucker 218 gesendet werden, die graduell in ihrer Größe erhöht werden, bis eine Druckzeit von ungefähr 300 Millisekunden erreicht ist. Nachdem eine Druckzeit von 300 Millisekunden erreicht ist, wird der Testpunkt aufgezeichnet, wobei dann zum nächsten Testpunkt fortgeschritten wird. Dieser Prozeß wird wiederholt, bis alle Testpunkte getestet wurden.

Anzahl der Testwerte	Anzahl der Dimensionen im Ressourcen-Element	Testzeit
1	0 (Linie)	300 ms
5	1 (Glyph)	1500 ms
10	2 (Rechteck, Keil)	3000 ms
15	3 (GlyphBDN)	4500 ms

Tabelle 4. Testzeiten für die Kennzeichnung der Elemente

Wenn das Druckermodell 256 die Kostendaten wir oben beschrieben erstellt, werden die Kostendaten in der Protokolldatei 260 gespeichert. Um den Datenabrufprozeß zu vereinfachen, werden die Kostendaten in der Protokolldatei 260 klassifiziert und in den Kostentabellen 262 gespeichert. Der Klassifikationsprozeß vereinfacht die für das Speichern der Kostendaten erforderliche Datenstruktur. Die vorliegende Ausführungsform macht mehrere Annahmen, die den Klassifikationsprozeß vereinfachen. In vielen Fällen ist es bekannt, daß derselbe Code im Drucker verwendet wird, um verschiedene Elemente zu zeichnen. Zum Beispiel wird derselbe Code, der das Zeichenelement GlyphB1 zeichnet, verwendet, um das Zeichenelement BitmapHR zu zeichnen. Wegen der Redundanz im Code braucht das System nur die Kostendaten für ein Element zu speichern. Einige nicht vom Ressourcenassembler 208 verwendeten Zeichenelemente, wie GlyphB1, müssen nicht gemessen werden.

Die Analyse der Punktkosten gibt auch an, daß die Punktkosten für ein bestimmtes ROP und eine bestimmtes Brush bei einem Zeichenelement dieselben sind wie bei einen anderen. Das heißt, daß die Koten pro Punkt für das Zeichenelement Rect, ROP0 und ein schwarzes Brush dieselben sind wie pro Punkt für das Zeichenelement Glyph, ROP0 und eine schwarzes Brush. Die Punktkosten unterscheiden sich also nicht für jedes Zeichenelement, sondern für jede Kombination aus ROP und Brush. Bei vier Brushes und 256 ROPs gibt es 1024 mögliche Punktkosten.

Um die Klassifikation zu vereinfachen, verwendet das System ein Gruppierungsschema bei der Klassifikation der Elemente. Unter Gruppierung ist ein Prozeß zu verstehen, der die Anzahl der Dateneinträge durch das Zusammenfassen oder Gruppieren von Zeichenelementen mit ähnlichen Kosten reduziert. Die vorliegende Ausführungsform der Erfindung berechnet die Kosten der Elementparameter P, D und R für alle Zeichenelemente und gruppiert sie in größere Kategorien. Wenn Gruppen von ähnlichen Zeichenelementen, wie SelectL, SelectB und SelectS, drei verschiedene Kosten aufweisen, ist es lediglich erforderlich, die teuersten Kosten der drei aufzunehmen. Da diese Befehle relativ selten verwendet werden, ist die Abweichung von geringfügiger Bedeutung. Wenn verschiedene Formen eines Elements gegeben sind, ist es lediglich erforderlich, die teuerste Form zu klassifizieren und diese Kosten für alle anderen Formen zu verwenden. Dabei enthalten die Kostentabellen immer jeweils die höheren Kosten, so daß kein Druckerfehler auftritt, weil fälschlicherweise angenommen wird, daß der Drucker ein Band in Echtzeit verarbeiten kann. Es ist zu beachten, daß das Schema nur als ein Hilfsmittel zum Reduzieren des für die Protokolldatei erforderlichen Speichers angegeben wird. Es ist möglich, aber nicht praktisch, die tatsächlichen Kosten für alle möglichen Elemente aufzunehmen.

Die Protokolldatei 260 enthält die Kostenmetrikdaten für alle Parameter für jedes der Zeichenelemente. Als ein Beispiel des Gruppierungsprozesses soll angenommen werden, daß eine Gruppe von Zeichenelementen die folgenden Kosten für den Parameter D aufweisen: 1,2,2,2,5,5,6,9,10,10. An Stelle der zehn Dateneinträge kann das System drei Gruppen von jeweils 2, 6 und 10 bilden. Die erste vier Einträge werden der ersten Gruppe mit einem Kostenaufwand von 2 zugeordnet, die nächsten drei Zeichenelemente werden der zweiten Gruppe mit einem Kostenaufwand von 6 zugeordnet, und die letzten drei Zeichenelemente werden der dritten Gruppe mit einem Kostenaufwand von 10 zugeordnet. Es ist zu beachten, daß daraus eine gewisse Ungenauigkeit resultiert, da einige Zeichenelemente tatsächliche Kosten aufweisen, die niedriger sind als die Kosten der Gruppe, der sie zugeordnet sind. Die Anzahl der Gruppen ist jedoch variabel und wird so gewählt, daß die Fehlergröße minimiert wird. Es ist weiterhin zu beachten, daß ein

Zeichenelement niemals einer Gruppe zugeordnet wird, die niedrigere Kosten angibt. Das Unterbewerten der tatsächlichen Kosten kann einen Druckerfehler verursachen, wenn der Ressourcenassembler 208 aufgrund der unterbewerteten tatsächlichen Kosten fälschlicherweise bestimmt, daß eine RPL in Echtzeit verarbeitet werden kann. Die praktische Erfahrung zeigt, daß 1024 Kostendateneinträgen für die Punktkosten durch nur 40 Gruppen von Punktkostendaten wiedergegeben werden können.

Der Gruppierungsverfahren wird durch das Anordnen aller Kosten für einen bestimmten Parameter in einer Reihe vorgenommen, wobei die Reihe von den niedrigsten zu den höchsten Kosten geordnet ist. Jeder Kostendateneintrag wird von seinem Nachbarn subtrahiert, und die Differenz wird in einer Matrix gespeichert. Ein Verschiebewert für jeden Matrixeintrag verfolgt die tatsächlichen Kosten. Dann wird die Differenz-Matrix von den größten Kosten zu den kleinsten Kosten sortiert. Der erste Eintrag in der sortierten Matrix entspricht den Gruppen-Datenkosten für eine Gruppe. Der zweite Eintrag in der sortierten Matrix entspricht den Gruppenkosten, wenn zwei Gruppen verwendet werden. Der dritte Eintrag in der sortierten Matrix entspricht den Gruppenkosten, wenn drei Gruppen verwendet werden usw.

Als ein Beispiel für das Gruppierungsverfahren sollen die folgenden Kosten für den Parameter D angenommen werden: 1,2,2,2,5,5,6,9,10,10. Wenn jeder Kostenaufwand von seinem Nachbarn subtrahiert wird, erhält man die folgende Matrix: 1(2-1), 0(2-2), 0(2-2), 3(5-2), 0(5-5), 1(6-5), 3(9-6), 1(10-9) und 0(10-10). Diese Matrix wird dann in sequentieller Reihenfolge von den größten zu den kleinsten angeordnet, wobei die mit den höchsten Kosten assoziierte Differenzzahl als erste aufgelistet wird. In dem vorstehenden Beispiel ist die sequentiell angeordnete Differenz-Matrix wie folgt: 3,3,1,1,1,0,0,0,0. Die erste Zahl in der Differenz-Matrix ist mit der Kostendifferenz zwischen 9 und 6 assoziiert, während die zweite Zahl 3 mit der Kostendifferenz zwischen 5 und 2 assoziiert ist. Bei zwei Kostengruppen werden deshalb die Gruppen durch die mit der ersten Zahl in der sequentiellen Differenz-Matrix assoziierten Kostendifferenz geteilt. In diesem Beispiel werden die zwei Gruppen zwischen den mit der ersten Zahl 3 assoziierten Kostenzahlen 6 und 9

geteilt. Eine Gruppe enthält also die Kosten 9-10, währen die zweite Gruppe die Kosten 1-6 enthält. Wenn drei Gruppen verwendet werden, werden die zweite und die dritte Gruppe entsprechend zwischen den mit der zweiten Zahl 3 assoziierten Kostenzahlen 2 und 5 geteilt. Die Kostengruppe sind dann also 9-10, 5-6 und 1-2. Das Prinzip kann auf eine beliebige Zahl von Kostengruppen erweitert werden. Alternativ dazu können statistische Verfahren wie eine Histogrammanalyse verwendet werden, um die geeignetsten Gruppenkosten zu bestimmen.

Die gruppierten Dateneinträge werden in den Kostentabellen 262 gespeichert. Eine entsprechende Map-Tabelle 264 wird verwendet, um den Zugriff auf die Kostentabelle zu erleichtern. Die 40 Punktkostentabellen werden zum Beispiel in einer Kostentabelle mit relativen Lokationen von 0 bis 39 gespeichert. Dabei ist zu beachten, daß die Kostentabellen in der Praxis an einer beliebigen geeigneten Lokation im Hostcomputer 202 gespeichert sein können. Die Punktkostentabelle enthält 1024 Einträge (4x256), die jeweils eine Zahl zwischen 0 und 39 enthalten. Der Eintrag in der Map-Tabelle 264 entspricht einem Eintrag in der Kostentabelle, die Teil der Kostentabellen 264 ist.

Der Ressourcenassembler 208 verwendet die Map-Tabellen 264 und die Kostentabellen 262, um die Kosten für jedes Zeichenelement zu bestimmen. In einer Prozedur kann der Ressourcenassembler 208 jedesmal auf die Punktkostentabellen zugreifen, wenn eine Operation mit einem Zeichenelement Punktkosten verursacht. Es können Gesamtkosten akkumuliert werden. Die Gesamtkosten entsprechen dem Kostendateneintrag in der Kostentabelle mal der Anzahl der Punkte des besonderen Typs. Das derartige Berechnen der Gesamtkosten erfordert jedoch eine Multiplikationsoperation für jedes Zeichenelement in einem Band. Wie dem Fachmann und auch dem Laien wohlbekannt ist, sind Multiplikationsoperationen zeitaufwendig. Deshalb verwendet die vorliegende Ausführungsform 40 Punktzähler für jedes Band (ein Punktzähler für jede der 40 gruppierten Punktkosten). Immer wenn eine Operation einen Punkt betrifft, erhöht der Ressourcenassembler 208 eine der entsprechenden 40 Punktzähler. Am Ende jedes Bandes multipliziert der Ressourcenzähler 208 die Punktsumme in jedem der 40 Punktzähler mit dem Kostendateneintrag in

der Kostentabelle 262 für den entsprechenden Punkt. Dann werden die vierzig multiplizierten Werte summiert, um die Gesamtpunktkosten für das Band zu erhalten. Diese Prozedur umfaßt weniger Multiplikationen als die erste Prozedur. Es ist auch zu beachten, daß einige der vierzig Punktzähler in einem gegebenen Band einen Wert von 0 aufweisen kann, der geprüft werden kann, bevor eine Multiplikationsoperation durchgeführt wird. Dies spart zusätzliche Zeit bei der Berechnung der Gesamtpunktkosten. Entsprechend werden die anderen Kosten für die Zeichenelemente berechnet, um die Gesamtkosten für ein Band zu erhalten.

Der Ressourcenassembler 208 bestimmt die Gesamtkosten für das Band und vergleicht die Gesamtkosten mit der Zeit, die für das Verarbeiten des Bandes mit dem Drucker 218 verfügbar ist. Wenn der Drucker über genug Zeit verfügt, um das Band zu verarbeiten, werden die RPL-Daten für dieses Band nicht durch den Hostcomputer 202 verarbeitet. Wenn das Band zu komplex für die Echtzeit-Verarbeitung ist, wird das Band als ein möglicher Kandidat für die Vorverarbeitung durch den Drucker 218 markiert. Wenn die Kosten für die gesamte Seite durch den Ressourcenassembler unter Verwendung der oben beschriebenen Kostenmetrik-Prozeduren berechnet wurden, werden die Bänder geprüft, um zu sehen, ob die Anzahl der komplexen Bänder eine vorbestimmte Zahl überschreitet. Wie zuvor genannt, hängt die Anzahl der komplexen Bänder, die als „zu groß" betrachtet wird, von zahlreichen Faktoren, wie der Berechnungsleistung des Druckers, der Geschwindigkeit der Druckermechanismus, den Überlastungskosten des Druckers und der für das Speichern der Banddaten verfügbaren Speichermenge des Druckerspeichers 222 (siehe Fig. 2) ab. Wenn relativ komplexe Bänder auf einer gegebenen Seite vorhanden sind, werden die komplexen Bänder für eine Vorverarbeitung zum Drucker 218 gesendet. Auf diese Weise werden alle komplexen Bänder vorverarbeitet, bevor der Drucker eine Echtzeit-Aktivierung des Druckermechanismus 226 vornimmt. Der Ressourcenassembler 208 kann auch einige der komplexen Bänder verarbeiten und die Bitmapdaten für diese Bänder zum Drucker 218 senden. Wenn zu viele komplexe Bänder vorhanden sind, verarbeitet der Ressourcenassembler 208 die gesamte Seite.

Als ein Beispiel für das Kostenberechnungsverfahren soll angenommen werden, daß ein Rechteck an einer bestimmten Stelle auf der Seite unter Verwendung von ROP0 und einem schwarzen Brush gezeichnet werden soll. Das Rechteck soll eine Breite w von 10 und eine Höhe h von 10 aufweisen. Der folgende Satz von Zeichenelementen kann verwendet werden, um ein derartiges Rechteck zu bilden: SetRowAbs(...), SetColAbs(...), SetExtAbs(...), SetRopandBruch(...) und Rect(w,h). Der Einfachheit halber wurden die Werte der Argumente für die meisten der Zeichenelemente nicht angegeben.

Die folgenden Schritte werden verwendet, um die Gesamtkosten für diese RPL zu erhalten:

1. Finden der Kostenmetrik für die ersten drei Zeichenelemente durch Zugreifen auf die Gruppentabelle und Erhalten der Werte P, R und D für die spezifizierten Parameter. Addieren der Kosten von P, R und D. Die Werte für diese Elemente sind die folgenden: Für **SetRowAbsS** (...) = 123,647 Mikrosekunden, **SetColAbs** (...) = 23,665 Mikrosekunden und **SetExtAbsS** (...) = 23,664 Mikrosekunden.

2. Finden der Kostenmetrik für das durch die SetRopAndBrush-Kombination spezifizierte ROP (ROP Nummer 41). Die Werte sind 224,633 Mikrosekunden für P, 11,292 Mikrosekunden für R und 0,04 Mikrosekunden für D. Die Gesamtkosten der Kombination aus ROP und Brush sind 341,6 Mikrosekunden (nach 341,6 Mikrosekunden sind die Bits für das Rechteck im Bildpuffer für das Drucken bereit).

Dabei ist zu beachten, daß der Treiber die in der BID-Tabelle nachgeschlagenen Kombinationen aus ROP und Brush verfolgt. Der zuvor verwendete Wert für die Kombination wird nicht erneut nachgeschlagen.

3. Addieren der Kosten der Kombination aus ROP und Brush zu den Kosten für alle Zeichenelemente in der RPL-Liste: 341,6 + 123, 647 + 23,665

+ 23,664 = 512,576 Mikrosekunden. Dies entspricht den Gesamtkosten der RPL.

Da die Ressourcen verwendet werden, um Daten zu drucken, ist die Kostenmetrik besonders nützlich für die Drucktechnologie. Der Hostcomputer 202 erzeugt die Ressourcen sowie die Abhängigkeiten der Ressourcen. Durch das Übersetzen der Ressourcen in Zeichenelemente können die Kosten für das Ausführen jedes Zeichenelement durch den Hostcomputer 202 berechnet werden. Das hier beschriebene System erlaubt dem Ressourcenassembler 208 einen einfachen Zugriff auf die Kostendaten in Laufzeit. Die für das Vorverarbeiten durch den Drucker 218 verfügbare Zeit wird in den Kostenmetrikdaten berechnet. Der Ressourcenassembler 208 verwendet die Kostenmetrikdaten, um zu entscheiden, welcher Teil des Computer-Drucker-Systems 200 die Daten verarbeiten sollte. Die Verwendung der Kostenmetrikdaten erlaubt es dem Hostcomputer 202 und dem Drucker 218 sich die Datenverarbeitungsverantwortlichkeiten in einer Weise zu teilen, wie es bisher nicht möglich war.

Die vorstehende Beschreibung sieht detaillierte Erläuterungen zu der Anwendung der erfinderischen Systems und Verfahrens vor, die den Druckerbetrieb analysieren. Es ist jedoch möglich, dieselben Techniken für die Analyse des auf das Drucken bezogenen Betriebs des Computers zu verwenden. Das erfinderische System und Verfahren sind auch für andere Anwendungen als das Drucken nützlich, wie zuvor erläutert wurde.

Es wurden verschiedene Ausführungsformen und Vorteile der vorliegenden Erfindung in der vorausgehenden Beschreibung auseinandergesetzt, wobei jedoch zu beachten ist, daß die obige Beschreibung lediglich illustrativ ist und daß Detailveränderungen vorgenommen werden können, die dennoch in den Grundprinzipien der vorliegenden Erfindung enthalten sind. Die vorliegende Erfindung ist lediglich durch die beigefügten Ansprüche beschränkt.

EPA 93 111 048.0

Patentansprüche

1. Computer-Drucker-System, das einen Drucker und einen Hostcomputer umfaßt, zum Steuern und Drucken eines Dokuments auf dem Drucker, wobei der Hostcomputer einen Ressourcenspeicherbereich zum Speichern von Ressourcen, eine Vielzahl von im Ressourcenspeicherbereich gespeicherten Ressourcen und eine Datendatei für das Dokument, die Daten enthält, die eine Vielzahl von zu druckenden Objekten beschreiben, umfaßt und wobei der Drucker einen Druckermechanismus umfaßt, wobei das System umfaßt:

einen Hostressourcenspeicher im Hostcomputer, der einen ausgewählten Ressourcensatz speichert, der eine Vielzahl von für das Drucken erforderlichen Ressourcen umfaßt,

einen Ressourcenassembler im Hostcomputer, der die Datendatei untersucht und wenigstens einige der Ressourcen aus dem Ressourcenspeicherbereich auswählt, um einen Teilsatz von ausgewählten, für das Drucken des Dokuments erforderlichen Ressourcen zu bilden, den Teilsatz von ausgewählten Ressourcen in den Hostressourcenspeicher lädt, die Datendatei in einen Satz von Elementen übersetzt, die der Vielzahl von Objekten in einem bestimmten Teil des Dokuments entsprechen, die Verarbeitungskosten für das Verarbeiten des Elementsatzes zu einer Bitmapdatei im Drucker berechnet und eine Zeitauslösung setzt, die wenigstens so groß ist wie die Verarbeitungskosten,

einen Druckerressourcenspeicher im Drucker, der den Teilsatz von ausgewählten Ressourcen und den Elementsatz aus dem Hostcomputer empfängt und den Teilsatz von ausgewählten Ressourcen und den Elementsatz speichert,

einen Ressourcenplaner im Drucker, der die Übertragung des Teilsatzes von ausgewählten Ressourcen und des Elementsatzes zum Druckerressourcenspeicher steuert, wobei der Ressourcenplaner während der Übertragung des Teilsatzes von ausgewählten Ressourcen und des Elementsatzes die Kommunikation mit dem Hostcomputer aufrechterhält, und

einen Ressourcenausführer im Drucker, der die Bitmapdatei in Übereinstimmung mit dem Teilsatz von ausgewählten Ressourcen und mit dem Elementsatz erzeugt und die Bitmapdatei für das Drucken zum Druckermechanismus überträgt, wobei die Zeitauslösung der tatsächlichen Verarbeitungszeit entspricht.

2. System zum dynamischen Verändern einer Drucker-Zeitauslösung, wenn ein Dokument auf einem Drucker von einem mit dem Drucker verbundenen Hostcomputer gedruckt wird, wobei der Hostcomputer eine Datendatei für das Dokument aufweist, die Daten enthält, die eine Vielzahl von zu druckenden Objekten beschreiben, wobei der Drucker einen Druckerressourcenspeicher aufweist, der die vom Hostcomputer empfangenen Daten empfängt und speichert, wobei das System umfaßt:

einen Ressourcenassembler im Hostcomputer, der die Datendatei untersucht und die Datendatei in einen Satz von Elementen übersetzt, die der Vielzahl von Objekten in einem bestimmten Teil des Dokuments entsprechen, die Verarbeitungskosten für das Verarbeiten des Elementsatzes in eine Bitmapdatei berechnet und eine Zeitauslösung setzt, die den Verarbeitungskosten entspricht, und

einen Ressourcenplaner im Drucker, der die Übertragung des Teilsatzes von ausgewählten Ressourcen und des Elementsatzes zum Druckerressourcenspeicher steuert, wobei die Zeitauslösung der tatsächlichen Verarbeitungszeit entspricht.

3. System zum dynamischen Verändern einer Drucker-Zeitauslösung, wenn ein Dokument auf einem Drucker von einem mit dem Drucker verbundenen Hostcomputer gedruckt wird, wobei der Hostcomputer einen Ressourcenspeicherbereich

zum Speichern von Ressourcen, eine Vielzahl von im Ressourcenspeicherbereich gespeicherten Ressourcen und eine Datendatei für das Dokument umfaßt, die Daten enthält, die eine Vielzahl von zu druckenden Objekten beschreiben, wobei der Drucker einen Ressourcenausführer umfaßt, der eine Bitmapdatei erzeugt, die den vom Hostcomputer übertragenen Daten entspricht, wobei das System umfaßt:

einen Hostressourcenspeicher im Hostcomputer, der einen ausgewählten Satz von Ressourcen speichert, der eine Vielzahl von für das Drucken des Dokuments erforderlichen Ressourcen umfaßt, und

einen Ressourcenassembler im Hostcomputer, der die Datendatei untersucht und wenigstens einige der Ressourcen aus dem Ressourcenspeicherbereich auswählt, um einen Teilsatz von ausgewählten Ressourcen zu bilden, der für das Drucken des Dokuments erforderlich ist, den Teilsatz von ausgewählten Ressourcen in den Hostressourcenspeicher lädt, die Datendatei in einen Satz von Elementen übersetzt, die der Vielzahl von Objekten in einem bestimmten Teil des Dokuments entsprechen, die Verarbeitungskosten für das Verarbeiten des Elementsatzes in eine Bitmapdatei im Drucker berechnet und eine Zeitauslösung setzt, die wenigstens so groß ist wie die Verarbeitungskosten.

4. System zum dynamischen Verändern einer Zeitauslösung bei der Kommunikation zwischen einem Peripheriegerät und einem mit dem Peripheriegerät verbundenen Hostcomputer, wobei das System umfaßt:

eine Datendatei im Hostcomputer, und

einen Ressourcenassembler im Hostcomputer, der die Datendatei untersucht, die Verarbeitungskosten des Peripheriegerätes für das Verarbeiten wenigstens eines Teils der Datendatei berechnet und eine Zeitauslösung setzt, die den Verarbeitungskosten entspricht.

5. System nach Anspruch 1, das weiterhin eine Protokolldatei im Hostcomputer umfaßt, wobei die Protokolldatei Kosteninformation für eine Vielzahl von Elementen speichert, wobei der Ressourcenassembler die Protokolldatei verwendet, um die Kosten für den Elementsatz zu bestimmen.

6. System nach Anspruch 5, das weiterhin einen Satz von Kostentabellen zum Speichern der klassifizierten Kostendaten für die Vielzahl von Elementen umfaßt, wobei jedes aus der Vielzahl von Elementen in Übereinstimmung mit der Kosteninformation klassifiziert ist, um die Speicheranforderungen der Protokolldatei zu reduzieren und um das Verfahren zum Bestimmen der Verarbeitungskosten durch den Ressourcenassembler zu vereinfachen.

7. System nach Anspruch 6, das weiterhin einen Satz von Map-Tabellen für die Vielzahl von Elementen umfaßt, wobei jede Map-Tabelle einen Zeiger auf eine Speicherstelle in der Kostentabelle vorsieht, an der die Kostendaten für jedes aus der Vielzahl von Elementen gespeichert sind.

8. System nach wenigstens einem der vorstehenden Ansprüche 1, 2, 3 und 4, das weiterhin eine Meßeinrichtung im Hostcomputer umfaßt, um eine Zeitauslösungs-Fehlermeldung zu erzeugen, wenn der Drucker länger als die Zeitauslösung außer Kommunikation ist.

9. System nach wenigstens einem der vorstehenden Ansprüche 1, 2 und 3, das weiterhin eine Einrichtung zum Verändern der Zeitauslösung von einem Teil des Dokuments zum nächsten in Abhängigkeit von den Verarbeitungskosten umfaßt.

10. System nach Anspruch 4, das weiterhin eine Einrichtung zum Verändern der Zeitauslösung von einem ersten Teil zu einem zweiten Teil der Datendatei in Abhängigkeit von den Verarbeitungskosten jeweils für den ersten und für den zweiten Teil umfaßt.

11. Verfahren in einem Computer-Drucker-System, das einen mit einem Hostcomputer verbundenen Drucker umfaßt, um eine Drucker-Zeitauslösung dynamisch zu verändern, wenn ein Dokument vom Hostcomputer auf dem Drucker gedruckt wird, wobei der Hostcomputer einen Ressourcenspeicherbereich zum Speichern von Ressourcen, eine Vielzahl von im Ressourcenspeicherbereich gespeicherten Ressourcen und eine Datendatei für das Dokument umfaßt, die Daten enthält, die eine Vielzahl von zu druckenden Objekten beschreiben, und wobei der Drucker einen Ressourcenausführer umfaßt, der eine Bitmapdatei erzeugt, die den vom Hostcomputer übertragenen Daten entspricht, wobei das Verfahren folgende Schritte umfaßt:

(a) Untersuchen der Datendatei und Auswählen von wenigstens einigen der Ressourcen aus dem Ressourcenspeicherbereich, um einen Teilsatz von ausgewählten Ressourcen zu bilden, die für das Drucken des Dokuments erforderlich sind,

(b) Übersetzen der Datendatei in einen Satz von Elementen, die der Vielzahl der in einem bestimmten Teil des Dokuments zu druckenden Objekten entsprechen,

(c) Berechnen der Verarbeitungskosten für den Ressourcenausführer und Verarbeiten des Elementsatzes zu einer Bitmapdatei im Drucker, und

(d) Setzen einer Zeitauslösung, die den Verarbeitungskosten entspricht, wobei die Zeitauslösung der tatsächlichen Verarbeitungszeit entspricht.

12. Verfahren nach Anspruch 11, das weiterhin folgende Schritte umfaßt:

(e) Übertragen des Teilsatzes von ausgewählten Ressourcen und des Elementsatzes vom Hostcomputer zum Drucker,

(f) Erzeugen der Bitmapdatei in Übereinstimmung mit dem Teilsatz von ausgewählten Ressourcen und dem Elementsatz, wobei der Drucker während der Erzeugung der Bitmapdatei außer Kommunikation mit dem Hostcomputer ist, und

(g) Übertragen der Bitmapdatei für das Drucken zum Druckmechanismus, wobei die Zeitauslösung der tatsächlichen Verarbeitungszeit entspricht.

13. Verfahren nach Anspruch 11, wobei der Schritt (c) zum Berechnen der Verarbeitungskosten einen Verarbeitungswert für jeden der besonderen Teile des Dokuments erzeugt und wobei der Schritt (d) zum Setzen der Zeitauslösung einen unterschiedlichen Wert für jeden der besonderen Teile des Dokuments in Abhängigkeit von dem Verarbeitungswert verwenden kann.

14. Verfahren nach Anspruch 11, das weiterhin einen Schritt zum Speichern der Kosteninformation für eine Vielzahl von Elementen in einer Protokolldatei umfaßt, wobei der Schritt (c) zum Berechnen der Verarbeitungskosten die Protokolldatei verwendet, um die Kosten für den Elementsatz zu bestimmen.

15. Verfahren nach Anspruch 14, das weiterhin einen Schritt zum Speichern von klassifizierten Daten für die Vielzahl von Elementen in einer Kostentabelle umfaßt, wobei jedes aus der Vielzahl von Elementen in Übereinstimmung mit der Kosteninformation klassifiziert wird, um die Speicheranforderungen der Protokolldatei zu reduzieren und um den Schritt (c) zum Berechnen der Verarbeitungskosten unter Verwendung der Protokolldatei zum Bestimmen der Kosten für den Elementsatz zu vereinfachen.

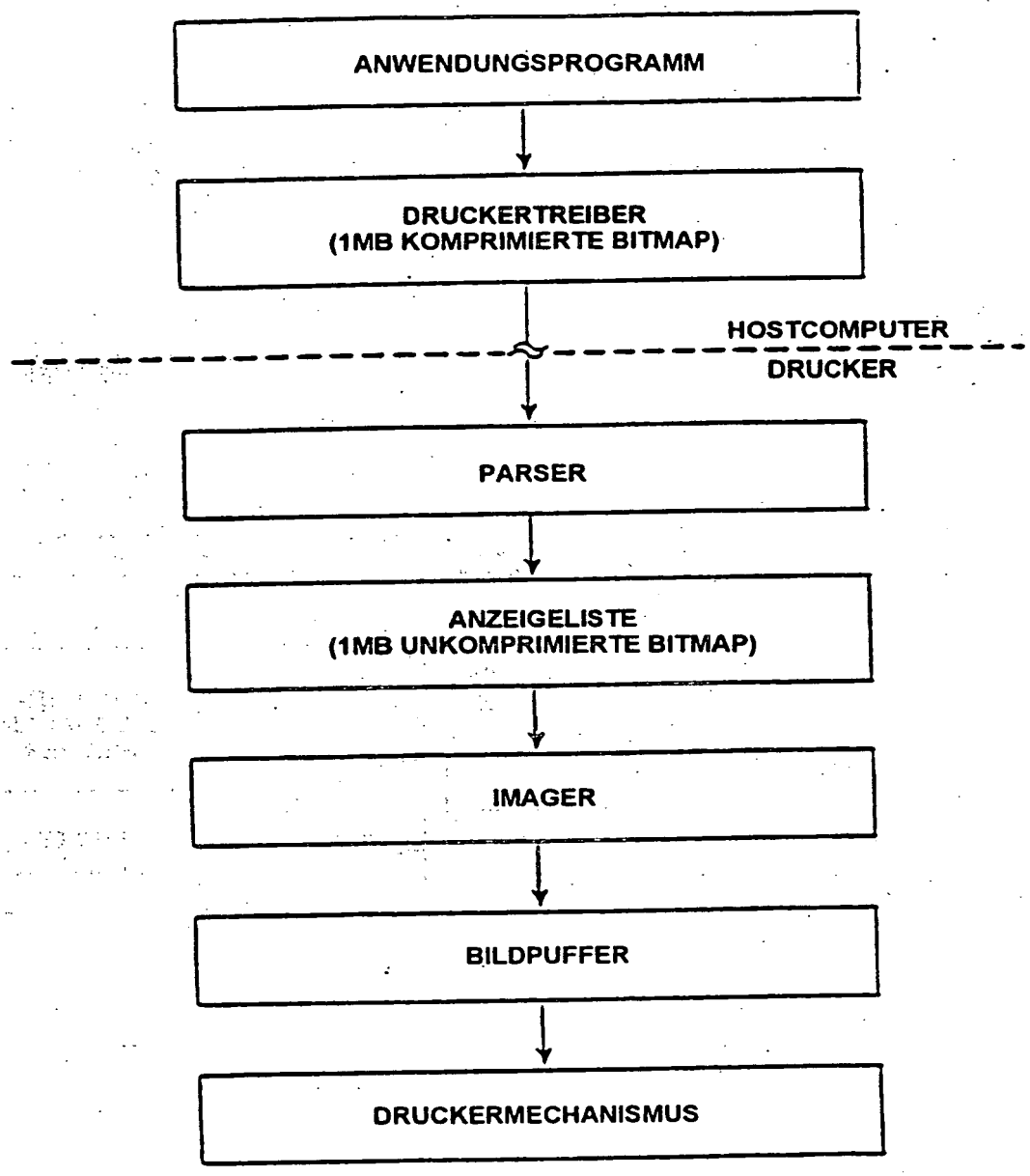
16. Verfahren in einem Computersystem, das ein mit einem Hostcomputer verbundenes Peripheriegerät umfaßt, um eine Zeitauslösung der Peripheriegeräts dynamisch zu verändern, wenn eine Datendatei in dem Peripheriegerät verarbeitet wird, wobei der Hostcomputer die Datendatei aufweist, wobei das Verfahren folgende Schritte umfaßt:

(a) Berechnen der Verarbeitungskosten zum Verarbeiten von wenigstens einem ersten Teil der Datendatei im Peripheriegerät, und

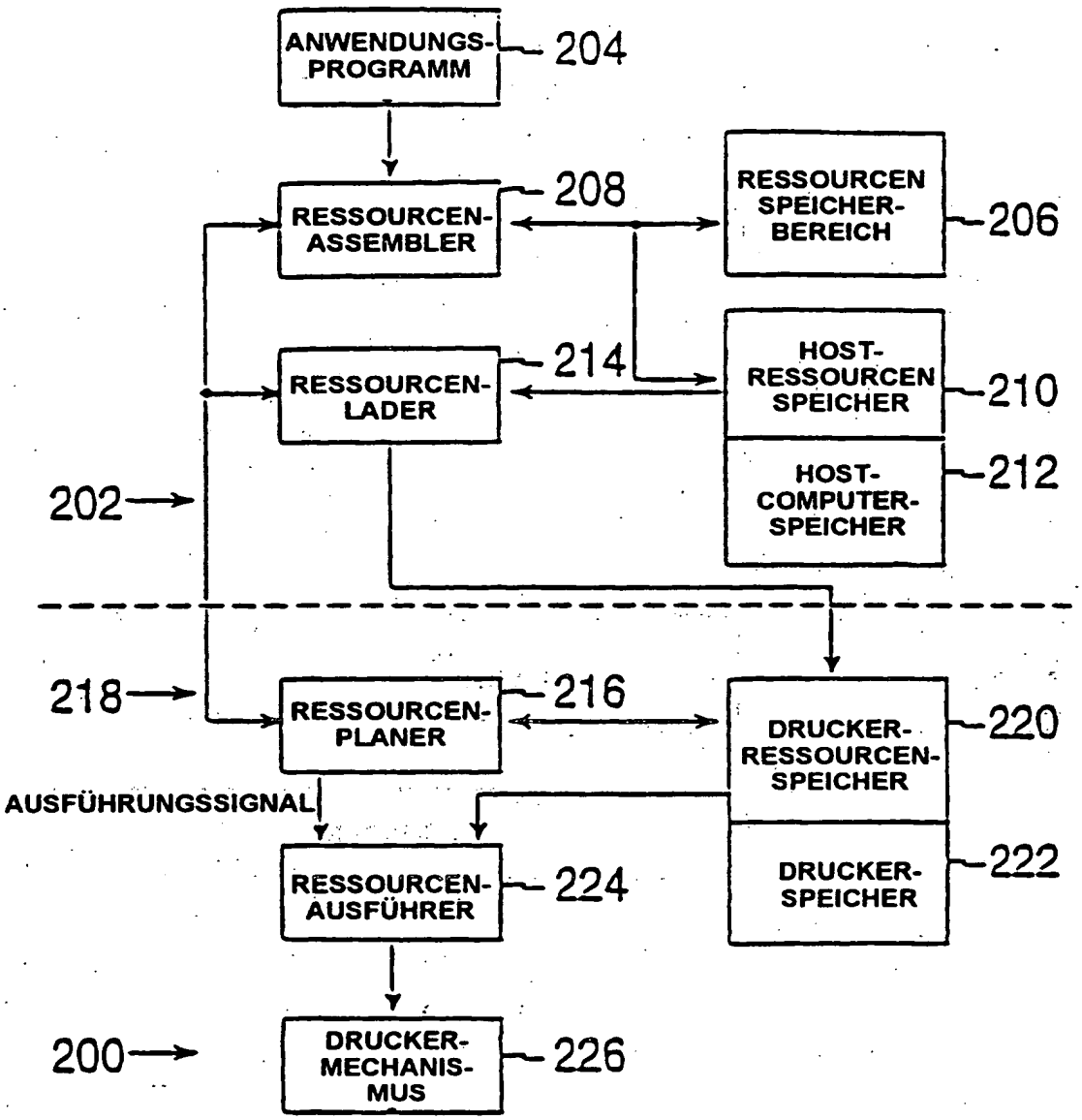
(b) Setzen einer Zeitauslösung, die den Verarbeitungskosten entspricht, wobei die Zeitauslösung der tatsächlichen Verarbeitungszeit entspricht.

17. Verfahren nach Anspruch 16, wobei der Schritt (a) zum Berechnen der Verarbeitungskosten die Verarbeitungskosten für einen zweiten Teil des Dokuments erstellt und der Schritt (b) zum Setzen der Zeitauslösung einen unterschiedlichen Wert für jeweils den ersten und den zweiten Teil des Dokuments in Abhängigkeit von den Verarbeitungskosten für jeweils den ersten und den zweiten Teil verwenden kann.

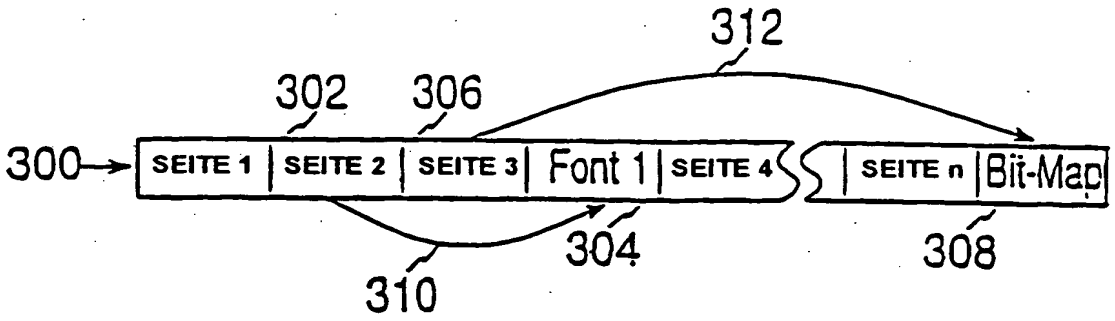
18. Verfahren nach Anspruch 11 oder 16, das weiterhin einen Schritt zum Erzeugen einer Zeitauslösungs-Fehlermeldung im Hostcomputer, wenn der Drucker länger als die Zeitauslösung außer Kommunikation ist, umfaßt.



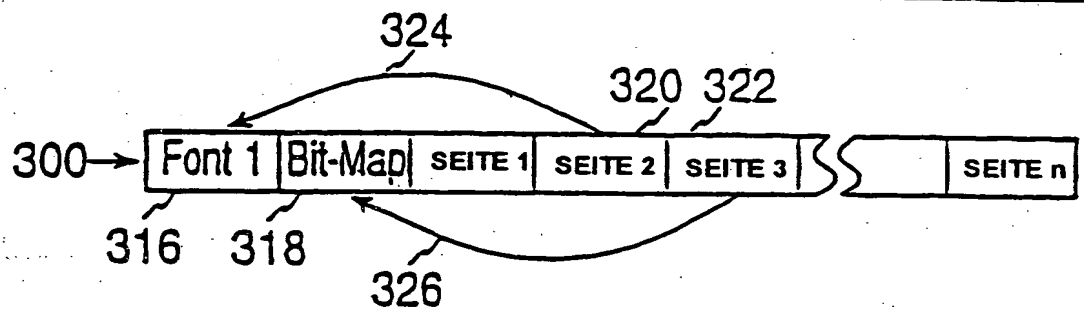
Figur 1 STAND DER TECHNIK


ANWENDUNGS-PROGRAMM
204
RESSOURCEN-ASSEMBLER
208
RESSOURCEN SPEICHER-BEREICH
206
RESSOURCEN-LADER
214
HOST-RESSOURCEN SPEICHER
210
HOST-COMPUTER-SPEICHER
212
202
218
RESSOURCEN-PLANER
216
DRUCKER-RESSOURCEN-SPEICHER
220
AUSFÜHRUNGSSIGNAL
RESSOURCEN-AUSFÜHRER
224
DRUCKER-SPEICHER
222
200
DRUCKER-MECHANIS-MUS
226

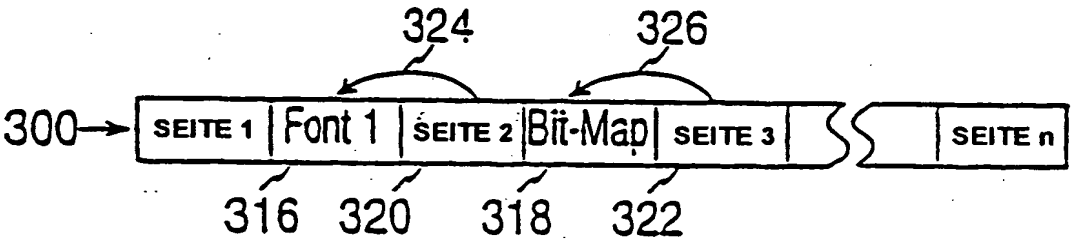
Figur 2



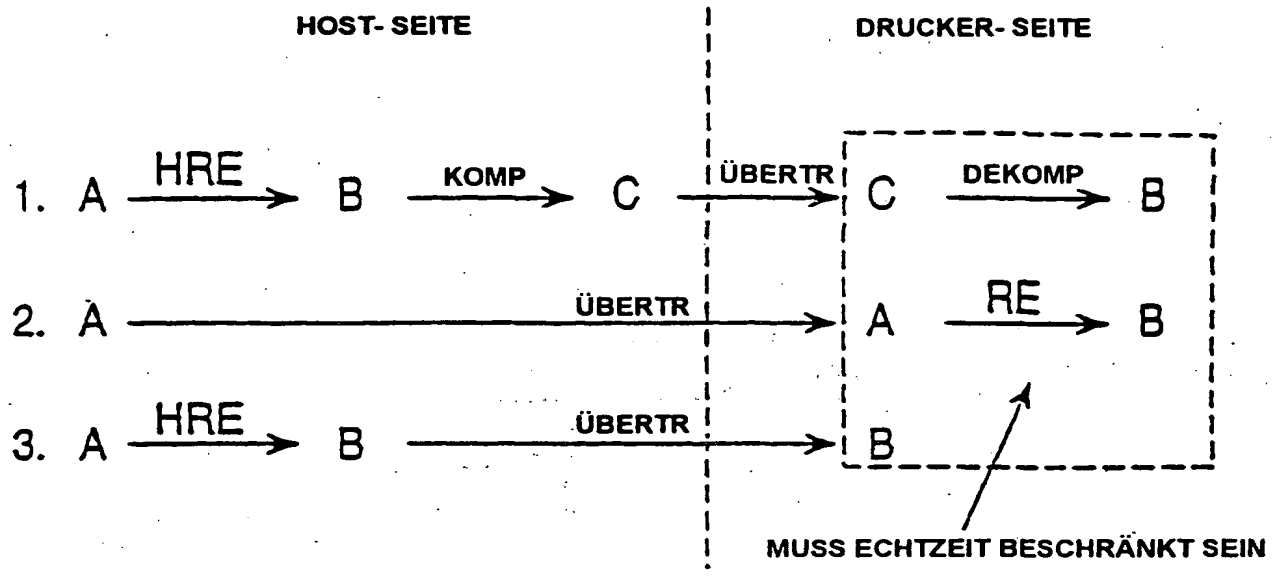
Figur 3A



Figur 3B

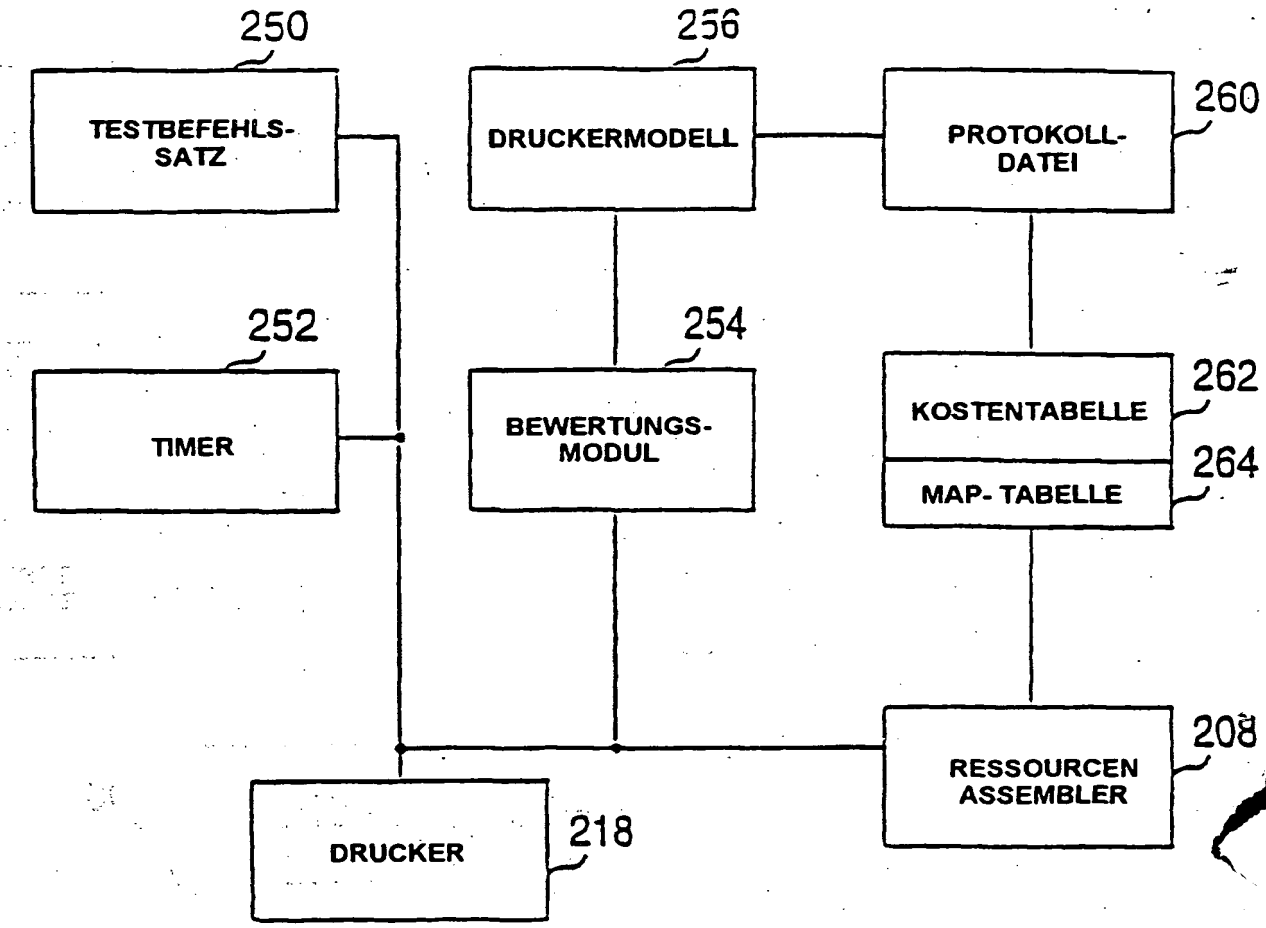


Figur 3C

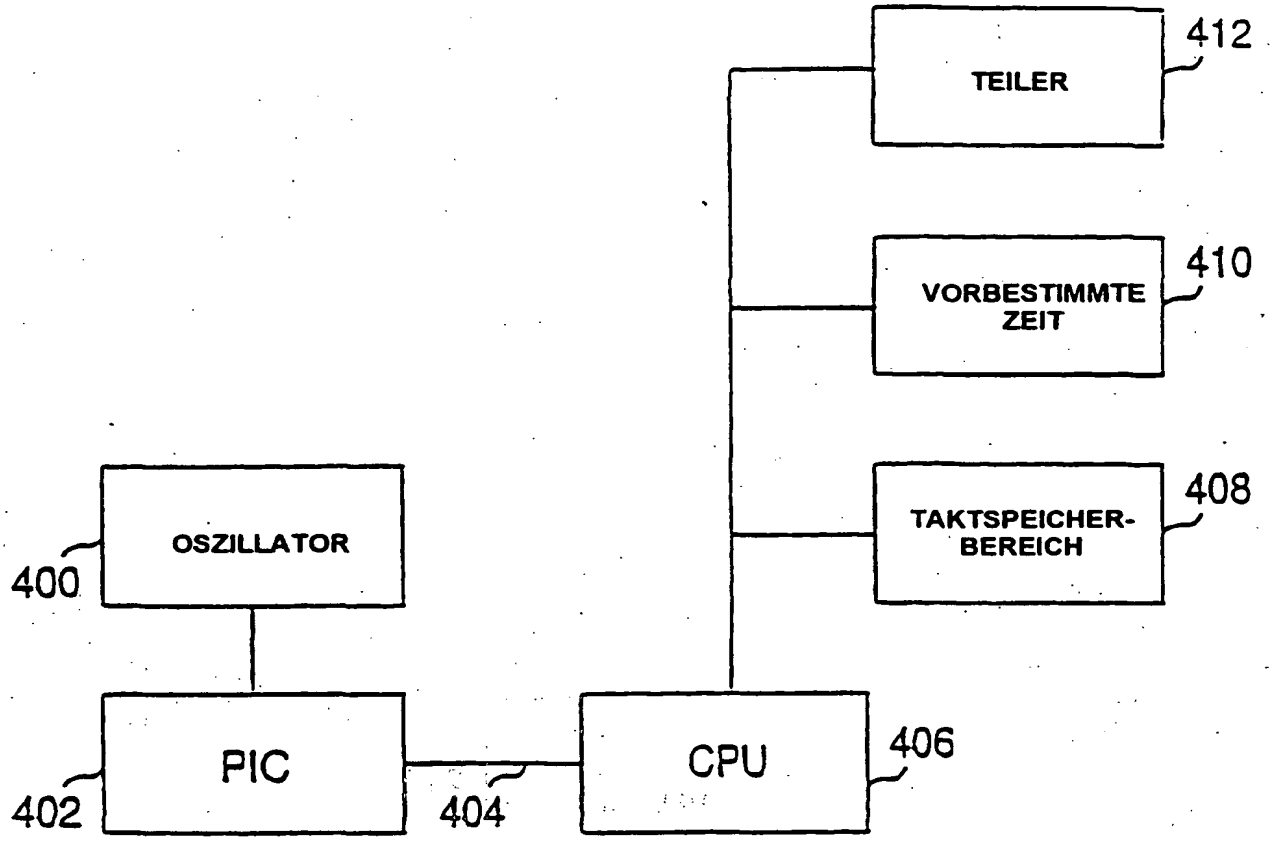


AUSGEWÄHLTE LASTEN- BALANCE OPTIONEN

Figur 4


250
TESTBEFEHLS-SATZ
256
DRUCKERMODELL
260
PROTOKOLL-DATEI
252
TIMER
254
BEWERTUNGS-MODUL
262
KOSTENTABELLE
264
MAP- TABELLE
208
RESSOURCEN ASSEMBLER
DRUCKER
218

Figur 5


TEILER
412
VORBESTIMMTE ZEIT
410
TAKTSPEICHER-BEREICH
408
OSZILLATOR
400
PIC
402
404
CPU
406

Figur 6